


```

000000    UU      UU      TTTTTTTTTT  PPPPPPPP    UU      UU      TTTTTTTTTT  HH      HH      LL      PPPPPPPP
000000    UU      UU      TTTTTTTTTT  PPPPPPPP    UU      UU      TTTTTTTTTT  HH      HH      LL      PPPPPPPP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP      PP
00      00    UU      UU      TT      PPPPPPPP    UU      UU      TT      HHHHHHHHHHHH  LL      PPPPPPPP
00      00    UU      UU      TT      PPPPPPPP    UU      UU      TT      HHHHHHHHHHHH  LL      PPPPPPPP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
00      00    UU      UU      TT      PP      PP    UU      UU      TT      HH      HH      LL      PP
000000    UUUUUUUUUUU  TT      PP      UUUUUUUUUUU  TT      HH      HH      LLLLLLLLLLL  PP
000000    UUUUUUUUUUU  TT      PP      UUUUUUUUUUU  TT      HH      HH      LLLLLLLLLLL  PP

```

```

LL                      IIIIII                      SSSSSSSS
LL                      IIIIII                      SSSSSSSS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SSSSSS
LL                      II                      SSSSSS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LLLLLLLLLLLLLLLL      IIIIII                      SSSSSSSS
LLLLLLLLLLLLLLLL      IIIIII                      SSSSSSSS

```



```
1 0001 0 MODULE lbr_outputhelp (
2 0002 0     LANGUAGE (BLISS32),
3 0003 0     IDENT = 'V04-000'
4 0004 0 ) =
5 0005 1 BEGIN
6 0006 1 %TITLE 'Prompting and library searching help function';
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 *   ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 *   TRANSFERRED.
20 0020 1 *
21 0021 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 *   CORPORATION.
24 0024 1 *
25 0025 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: Library access procedures.
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 LBR$OUTPUT_HELP outputs help text to a user specified output
38 0038 1 routine. This information is drawn from an explicitly named
39 0039 1 help library, or optionally, from user specified default help
40 0040 1 libraries. In addition, an optional prompting mode is available
41 0041 1 so that LBR$OUTPUT_HELP can interact with a user and continue to
42 0042 1 provide help information to a user after it has satisfied his
43 0043 1 initial help request.
44 0044 1
45 0045 1 ENVIRONMENT:
46 0046 1
47 0047 1 VAX native, user mode.
48 0048 1
49 0049 1 --
50 0050 1
51 0051 1
52 0052 1 AUTHOR: Peter George,          CREATION DATE: 01-May-1981
53 0053 1
54 0054 1 MODIFIED BY:
55 0055 1
56 0056 1 V03-005 GJA0081          Greg Awdziewicz          10-Apr-1984
57 0057 1 - Reduce the severity of the signaled "openin" error
```

58 0058 1
59 0059 1
60 0060 1
61 0061 1
62 0062 1
63 0063 1
64 0064 1
65 0065 1
66 0066 1
67 0067 1
68 0068 1
69 0069 1
70 0070 1
71 0071 1
72 0072 1
73 0073 1
74 0074 1
75 0075 1
76 0076 1
77 0077 1
78 0078 1
79 0079 1
80 0080 1
81 0081 1
82 0082 1
83 0083 1
84 0084 1
85 0085 1
86 0086 1
87 0087 1
88 0088 1
89 0089 1
90 0090 1 --

from fatal to error in the open library routine.
- Return any error from open library to the calling
program if the open fails for the specified main
library.
- Update the "ROUTINE VALUE:" comments to include the
codes "openin", "nohlplibs", and "usrinperr".

V03-004 GJA0077 Greg Awdziewicz 19-Feb-1984
- Set prompt_flags[hcf\$y_more] when switching libraries
at the end of the search_libs routine to fix AccVio
which was occurring when the subprompt for a topic which
is in more than one library comes from the subtopic in
other than the main library when this subtopic was
requested at the subtopic level from the main library.
- Flag the end-topic case to further gate the output
of the additional libraries info.
- Allow filenames to be 39 characters long.
- Change the routine Make_upper_case to use a translate
table instead of conditionals inside a loop.

V03-003 JWT0066 Jim Teague 22-Nov-1982
Let RMS parse filenames.

V03-002 PCG0014 Peter George 06-Oct-1982
Modify library name substring search so that it only
checks for substrings that start from the beginning of
the file name.

V03-001 PCG0013 Peter George 01-Jul-1982
Make default search libraries work again by correcting
typographic error.


```

: 92      0091 1 LIBRARY 'SYSS$LIBRARY:STARLET';
: 93      0092 1 REQUIRE 'PREFIX';
: 94      0231 1 REQUIRE 'LBRDEF';
: 95      0822 1
: 96      0823 1 EXTERNAL ROUTINE
: 97      0824 1     lbr$close : ADDRESSING_MODE(GENERAL),      ! Close library
: 98      0825 1     lbr$ini_control : ADDRESSING_MODE(GENERAL), ! Initialize librarian
: 99      0826 1     lbr$get_help : ADDRESSING_MODE(GENERAL),    ! Get help text
100      0827 1     lbr$open : ADDRESSING_MODE(GENERAL);        ! Open library
101      0828 1
102      0829 1 FORWARD ROUTINE
103      0830 1     prompt_help,      ! Main prompting loop
104      0831 1     search_libs,      ! Search default libraries till help is found
105      0832 1     change_lib,      ! Change current help lib context to specified library
106      0833 1     switch_libname,   ! Update the library name descriptor
107      0834 1     tran_next_lib,    ! Get next user specified default help library
108      0835 1     open_library,     ! Open a help library
109      0836 1     close_library,    ! Close a help library
110      0837 1     setup_keys,       ! Set up help keys for lbr$get_help
111      0838 1     call_lbrhelp,     ! Call lbr$get_help to get help from a library
112      0839 1     output_driver,    ! Driver for the user supplied output routine
113      0840 1     libs_available,   ! Output list of default libraries available
114      0841 1     file_present,     ! Determine if file exists
115      0842 1     nohelp_log,       ! Log unrecognized help requests
116      0843 1     remove_last_key,  ! Remove the last help key in a list of keys
117      0844 1     remove_terminator, ! Remove the termination character at the end of a command
118      0845 1     make_upper_case; ! Make a string all upper case
119      0846 1
120      0847 1 EXTERNAL LITERAL
121      0848 1     lbr$_endtopic,
122      0849 1     lbr$_illoutrou,
123      0850 1     lbr$_illinrou,
124      0851 1     lbr$_nohlplib,
125      0852 1     lbr$_toomnyarg,
126      0853 1     lbr$_usrinperr;
127      0854 1
128      0855 1 LITERAL
129      0856 1     filename_length= 39,      ! Max filename length.
130      0857 1     main_libnumber = -2,      ! Library number of /LIBRARY specified library
131      0858 1     external_libnumber = -1;   ! Library number of non-default, non-/LIB library
132      0859 1
133      0860 1 OWN
134      0861 1     end_topic_flag: BYTE,      ! Flag aborts text.
135      0862 1     sys$help : COUNTEDSTRING ('SYSS$HELP:.HLB'), ! Default library spec
136      0863 1     topic : COUNTEDSTRING ('Topic? '),      ! Topic prompt string
137      0864 1     subtopic : COUNTEDSTRING ('Subtopic? '), ! Subtopic prompt string
138      0865 1     prompt_prefix : VECTOR [5, BYTE]        ! Prompt control characters
139      0866 1     INITIAL (BYTE (4, 13, 10, 13, 10)),
140      0867 1     control_flags : BBLOCK [hcf$_length];    ! Global control flags
141      0868 1
142      0869 1 BIND
143      0870 1     help_flags = control_flags [hcf$_userlib], ! User default library flags
144      0871 1     prompt_flags = control_flags [hcf$_prompt]: BLOCK[, BYTE]; ! Prompt control flags
145      0872 1
146      0873 1 MAP
147      0874 1     help_flags : BBLOCK;
```

```
149 0875 1 %SBTTL 'Routine lbr$output_help';
150 0876 1 GLOBAL ROUTINE lbr$output_help (output_rout, output_size, keys_desc,
151 0877 1 library_desc, flags, input_rout) =
152 0878 2 BEGIN
153 0879 2
154 0880 2 !++
155 0881 2 FUNCTIONAL DESCRIPTION:
156 0882 2
157 0883 2 This routine is the entry point for this module. It performs
158 0884 2 some initial processing on the input parameters, opens the main
159 0885 2 help library, and then calls prompt_help to do all the real work.
160 0886 2
161 0887 2 CALLING SEQUENCE:
162 0888 2
163 0889 2 status = LBR$OUTPUT_HELP (output_routine, output_width [, [keys_desc]
164 0890 2 [, [library_desc] [, [flags] [, [input_routine] ]]] )
165 0891 2
166 0892 2 INPUTS:
167 0893 2
168 0894 2 output_rout = address of user output routine
169 0895 2 output_size = width of the output line to be passed to the user
170 0896 2 output routine
171 0897 2 keys_desc = address of string desc for keys
172 0898 2 library_desc = address of string desc for help library name
173 0899 2 flags = address of longword of option flags
174 0900 2 input_rout = address of user input routine
175 0901 2
176 0902 2 OUTPUTS:
177 0903 2
178 0904 2 The requested help text is passed to the output routine.
179 0905 2
180 0906 2 ROUTINE VALUE:
181 0907 2
182 0908 2 status lbr$_normal
183 0909 2 lbr$_illoutrou Output routine improperly specified or missing
184 0910 2 lbr$_illinrou Input routine improperly specified or missing
185 0911 2 lbr$_toomnyarg Too many arguments
186 0912 2 lbr$_nohlplib No help libraries can be opened.
187 0913 2 lbr$_usrinperr Input error from user's action routine.
188 0914 2
189 0915 2 Some errors which occur may be signaled. For example, if an error
190 0916 2 occurs while trying to open the specified main default library
191 0917 2 the error will be signaled as an "openin" error. In this case
192 0918 2 the return code value will have the inhibit bit set. So the
193 0919 2 return value will be
194 0920 2
195 0921 2 shr$_openin OR hlp$_facility OR sts$_error OR sts$_inhib_msg
196 0922 2
197 0923 2
198 0924 2 MAP
199 0925 2 keys_desc : REF BBLOCK,
200 0926 2 library_desc : REF BBLOCK;
201 0927 2
202 0928 2 BUILTIN
203 0929 2 ACTUALCOUNT,
204 0930 2 NULLPARAMETER;
205 0931 2
```



```

: 206 0932 2 LOCAL
: 207 0933 2   getcmd_line : BBLOCK [hlp$c_pagesize],      ! Command line buffer
: 208 0934 2   getcmd_desc : BBLOCK [dsc$c_s_bln],      ! Command desc
: 209 0935 2   indices : BBLOCK [hli$c_length],         ! Help library indices
: 210 0936 2   input_routine,                          ! Address of user input routine
: 211 0937 2   librarystring : BBLOCK [nam$c_maxrss],   ! Default library name string
: 212 0938 2   libraryname : BBLOCK [dsc$c_s_bln],     ! String descriptor for library name
: 213 0939 2   nomsg,                                  ! Open library message flag
: 214 0940 2   output_routine,                         ! Address of user output routine
: 215 0941 2   output_width,                          ! Width of output line
: 216 0942 2   status;
: 217 0943 2
: 218 0944 2 BIND
: 219 0945 2   main_libindex = indices [hli$l_mainindex], ! Index of /LIB library
: 220 0946 2   last_libindex = indices [hli$l_lastindex], ! Index of last library examined
: 221 0947 2   last_libnumber = indices [hli$l_lastnumb]; ! No. of last lib examined, relative to all default
: 222 0948 2
: 223 0949 2   help_flags = 0;                          ! Clear help control flags
: 224 0950 2
: 225 0951 2   IF ACTUALCOUNT() GTR hlp$c_params      ! If too many arguments
: 226 0952 2   THEN RETURN lbr$_toomnyarg;              ! then return error
: 227 0953 2
: 228 0954 2   IF NULLPARAMETER (hlp$c_outrou)         ! If output_routine missing
: 229 0955 2   OR .output_rout EQL 0                    ! or zero
: 230 0956 2   THEN RETURN lbr$_illoutrou              ! then return error
: 231 0957 2   ELSE output_routine = .output_rout;      ! else store data
: 232 0958 2
: 233 0959 2   IF NULLPARAMETER (hlp$c_outwidth)       ! If output_width missing
: 234 0960 2   OR ..output_size EQL 0                  ! or zero
: 235 0961 2   THEN output_width = hlp$c_liswidth      ! then use default
: 236 0962 2   ELSE output_width = ..output_size;      ! else store data
: 237 0963 2
: 238 0964 2   getcmd_desc = 0;
: 239 0965 2   getcmd_desc [dsc$a_pointer] = getcmd_line;
: 240 0966 2   IF NOT NULLPARAMETER (hlp$c_linedesc)    ! If keys_desc present
: 241 0967 2   THEN BEGIN                               ! then pick up passed descriptor
: 242 0968 2       getcmd_desc [dsc$w_length] = .keys_desc [dsc$w_length];
: 243 0969 2       CH$MOVE (.getcmd_desc [dsc$w_length], ! fill buffer
: 244 0970 2       .keys_desc [dsc$a_pointer],
: 245 0971 2       .getcmd_desc [dsc$a_pointer] );
: 246 0972 2   END;
: 247 0973 2
: 248 0974 2   libraryname = 0;
: 249 0975 2   libraryname [dsc$a_pointer] = librarystring; ! Init pointer to library name buffer
: 250 0976 2   IF NOT NULLPARAMETER (hlp$c_libname)     ! If library_name specified
: 251 0977 2   THEN BEGIN                               ! then override default
: 252 0978 2       help_flags [hlp$v_library] = true;
: 253 0979 2       libraryname [dsc$w_length] = .library_desc [dsc$w_length];
: 254 0980 2       CH$MOVE (.libraryname [dsc$w_length],
: 255 0981 2       .library_desc [dsc$a_pointer],
: 256 0982 2       .libraryname [dsc$a_pointer]);
: 257 0983 2   END
: 258 0984 2   ELSE help_flags [hlp$v_library] = false;
: 259 0985 2
: 260 0986 2   IF NOT NULLPARAMETER (hlp$c_flags)         ! If flags present
: 261 0987 2   THEN help_flags = .help_flags              ! then get relevent bits
: 262 0988 2   OR (hlp$m_prompt AND ..flags)

```



```
263 0989 3 OR (hlp$m_process AND ..flags)
264 0990 3 OR (hlp$m_group AND ..flags)
265 0991 3 OR (hlp$m_system AND ..flags)
266 0992 3 OR (hlp$m_liblist AND ..flags)
267 0993 3 OR (hlp$m_help AND ..flags)
268 0994 2 ELSE help_flags = .help_flags OR hlp$m_prompt ! else set defaults
269 0995 2 OR hlp$m_process OR hlp$m_group
270 0996 2 OR hlp$m_system AND NOT hlp$m_liblist
271 0997 2 AND NOT hlp$m_help;
272 0998 2
273 0999 2 IF .help_flags [hlp$v_prompt] ! If prompting enabled
274 1000 3 THEN BEGIN
275 1001 4 IF ( NULLPARAMETER (hlp$c_inrout) ! And output_routine missing
276 1002 4 OR .input_rout EQL 0) ! or zero
277 1003 3 THEN RETURN lbr$_illinrou ! then return error
278 1004 3 ELSE input_routine = .input_rout; ! else store data
279 1005 3 prompt_flags = hcf$m_cont; ! Turn on prompting
280 1006 3 END
281 1007 2 ELSE prompt_flags = hcf$m_noprompt; ! else turn off prompting
282 1008 2
283 1009 2 IF .help_flags [hlp$v_process] OR ! If default lib searching enabled
284 1010 2 .help_flags [hlp$v_group] OR .help_flags [hlp$v_system]
285 1011 2 THEN help_flags = .help_flags OR hlp$m_all ! then set all flag
286 1012 2 ELSE help_flags = .help_flags AND NOT hlp$m_all; ! else clear all flag
287 1013 2
288 1014 2 IF .help_flags [hlp$v_library] ! If library specified
289 1015 3 THEN BEGIN ! Then open it
290 1016 3 LOCAL
291 1017 3 local_status;
292 1018 3 nomsg = false; ! Signal error if library can't be opened
293 1019 3 local_status = open_library (main_libindex, libraryname, .nomsg); ! Open main library
294 1020 3 IF NOT .local_status THEN RETURN .local_status; ! Return any errors.
295 1021 3 last_libindex = .main_libindex; ! Set last library used to main library
296 1022 3 last_libnumber = main_libnumber;
297 1023 3 END
298 1024 3
299 1025 3 ELSE BEGIN ! Else use a default library
300 1026 3 LOCAL
301 1027 3 libno,
302 1028 3 acmode;
303 1029 3
304 1030 3
305 1031 3 IF NOT .help_flags [hlp$v_all] ! Are we allowed to?
306 1032 3 THEN RETURN lbr$_nohlp[lib]; ! If not then signal error
307 1033 3 libno = -1; ! Initialize search
308 1034 3 status = false; ! Init while condition
309 1035 3 nomsg = true; ! Do not signal open errors
310 1036 3 WHILE NOT .status ! While more libraries
311 1037 4 DO BEGIN ! Get and try to open one
312 1038 4 IF NOT tran_next_lib (libraryname, acmode, libno)
313 1039 4 THEN RETURN lbr$_nohlp[lib];
314 1040 4 status = open_library (last_libindex, libraryname, .nomsg);
315 1041 4 END;
316 1042 3 last_libnumber = 0; ! Set lib number
317 1043 3 END;
318 1044 2
319 1045 2 !
```



```

: 320      1046 2 ! Call prompt_help to do the real help work.
: 321      1047 2 !
: 322      1048 2 status = prompt_help(getcmd_desc, output_width, .input_routine, .output_routine,
: 323      1049 2 indices, libraryname);
: 324      1050 2
: 325      1051 2 IF .help_flags [hlp$v-library] ! If library specified
: 326      1052 2 THEN close_library(main_libindex); ! Close the main library
: 327      1053 2
: 328      1054 2 RETURN .status
: 329      1055 2
: 330      1056 1 END; ! Of lbr$output_help
```

```
.TITLE LBR_OUTPUTHELP Prompting and library searching
help function
```

```
.IDENT \V04-000\
```

```
.PSECT $OWNS,NOEXE,2
```

```
00000 END_TOPIC_FLAG:
```

```
.BLKB 1
```

```
.BLKB 3
```

```
00001 SYSHELP:
```

```
.BYTE 13
```

```
00004 .ASCII \SYSSHELP:.HLB\
```

```
.BLKB 2
```

```
00012 TOPIC:
```

```
.BYTE 7
```

```
00014 .ASCII \Topic? \
```

```
.BLKB 10
```

```
00015 SUBTOPIC:
```

```
.ASCII \Subtopic? \
```

```
.BLKB 1
```

```
0001C PROMPT_PREFIX:
```

```
.BYTE 4, 13, 10, 13, 10
```

```
.BLKB 3
```

```
0002D CONTROL_FLAGS:
```

```
.BLKB 8
```

```
HELP_FLAGS=
PROMPT_FLAGS=
```

```
CONTROL_FLAGS
```

```
CONTROL_FLAGS+4
```

```
.EXTRN
```

```
LBR$CLOSE, LBR$INI CONTROL
```

```
.EXTRN
```

```
LBR$GET_HELP, LBR$OPEN
```

```
.EXTRN
```

```
LBR$ENDTOPIC, LBR$_ILLOUTROU
```

```
.EXTRN
```

```
LBR$_ILLINROU, LBR$_NOHLPLIBS
```

```
.EXTRN
```

```
LBR$_TOOMNYARG, LBR$_USRINPERR
```

```
.PSECT $CODES,NOVRT,2
```

```
.ENTRY
```

```
LBR$OUTPUT_HELP, Save R2,R3,R4,R5,R6,R7
```

```
MOVAB
```

```
HELP_FLAGS, R7
```

```
MOVAB
```

```
-808(SP), SP
```

```
CLRL
```

```
HELP_FLAGS
```

```
CMPB
```

```
(AP), #6
```

```
BLEQU
```

```
1$
```

```
MOVL
```

```
#LBR$_TOOMNYARG, R0
```

```
RET
```

```
(AP)
```

```
TSTB
```

```
2$
```

```
BEQL
```

```
2$
```

```
: 0876
```

```
: 0949
```

```
: 0951
```

```
: 0952
```

```
: 0954
```

```
:
```

		04	AC	D5	0001F	TSTL	4(AP)			
		08	12	00022	BNEQ	3\$				
		50	00000000G	8F	D0 00024	2\$: MOVL	#LBR\$_ILLOUTROU, R0	0956		
				04	0002B	RET				
		56	04	AC	D0 0002C	3\$: MOVL	OUTPUT_ROUT, OUTPUT_ROUTINE	0957		
		02		6C	91 00030	CMPB	(AP), #2	0959		
				0A	1F 00033	BLSSU	4\$			
			08	AC	D5 00035	TSTL	8(AP)			
				05	13 00038	BEQL	4\$			
			08	BC	D5 0003A	TSTL	@OUTPUT_SIZE	0960		
				07	12 0003D	BNEQ	5\$			
		08	AE	50	8F 9A 0003F	4\$: MOVZBL	#80, OUTPUT_WIDTH	0961		
				05	11 00044	BRB	6\$			
		08	AE	08	BC D0 00046	5\$: MOVL	@OUTPUT_SIZE, OUTPUT_WIDTH	0962		
				0120	CE D4 0004B	6\$: CLRL	GETCMD_DESC	0964		
		0124	CE	0128	CE 9E 0004F	MOVAB	GETCMD_LINE, GETCMD_DESC+4	0965		
			03		6C 91 00056	CMPB	(AP), #3	0966		
				17	1F 00059	BLSSU	7\$			
			0C	AC	D5 0005B	TSTL	12(AP)			
				12	13 0005E	BEQL	7\$			
			50	0C	AC D0 00060	MOVL	KEYS_DESC, R0	0968		
		0120	CE	60	B0 00064	MOVW	(R0), GETCMD_DESC			
		04	B0	0120	CE 28 00069	MOVC3	GETCMD_DESC, @4(R0), @GETCMD_DESC+4	0971		
				0C	AE D4 00072	7\$: CLRL	LIBRARYNAME	0974		
		10	AE	14	AE 9E 00075	MOVAB	LIBRARYSTRING, LIBRARYNAME+4	0975		
			04		6C 91 0007A	CMPB	(AP), #4	0976		
				1A	1F 0007D	BLSSU	8\$			
			10	AC	D5 0007F	TSTL	16(AP)			
				15	13 00082	BEQL	8\$			
		01	A7	04	88 00084	BISB2	#4, HELP_FLAGS+1	0978		
			50	10	AC D0 00088	MOVL	LIBRARY_DESC, R0	0979		
		0C	AE	60	B0 0008C	MOVW	(R0), LIBRARYNAME			
		10	BE	04	B0 0C	AE 28 00090	MOVC3	LIBRARYNAME, @4(R0), @LIBRARYNAME+4	0982	
				04	11 00097	BRB	9\$	0976		
		01	A7	04	8A 00099	8\$: BICB2	#4, HELP_FLAGS+1	0984		
			05		6C 91 0009D	9\$: CMPB	(AP), #5	0986		
				4D	1F 000A0	BLSSU	10\$			
			14	AC	D5 000A2	TSTL	20(AP)			
				48	13 000A5	BEQL	10\$			
		50	14	BC	00 EF 000A7	EXTZV	#0, #1, @FLAGS, R0	0988		
				50	67 C8 000AD	BISL2	HELP_FLAGS, R0			
		51	14	BC	FF FFFF FD	8F CB 000B0	BICL3	#-3, @FLAGS, R1	0989	
				51	50 C8 000B9	BISL2	R0, R1			
		50	14	BC	FF FFFF FB	8F CB 000BC	BICL3	#-5, @FLAGS, R0	0990	
				50	51 C8 000C5	BISL2	R1, R0			
		51	14	BC	FF FFFF F7	8F CB 000C8	BICL3	#-9, @FLAGS, R1	0991	
				50	50 C8 000D1	BISL2	R0, R1			
		50	14	BC	FF FFFF EF	8F CB 000D4	BICL3	#-17, @FLAGS, R0	0992	
				50	51 C8 000DD	BISL2	R1, R0			
		51	14	BC	FF FFFF DF	8F CB 000E0	BICL3	#-33, @FLAGS, R1	0993	
				51	50 C9 000E9	BISL3	R0, R1, HELP_FLAGS			
				03	11 000ED	BRB	11\$	0987		
		67		0F	88 000EF	10\$: BISB2	#15, HELP_FLAGS	0996		
		1C		67	E9 000F2	11\$: BLBC	HELP_FLAGS, 14\$	0999		
				06	6C 91 000F5	CMPB	(AP), #6	1001		
				05	1F 000F8	BLSSU	12\$			
			18	AC	D5 000FA	TSTL	24(AP)			

			08	12	000FD		BNEQ	13\$			
		50	00000000G	8F	D0	000FF	12\$:	MOVL	#LBR\$_ILLINROU, R0		1003
		54	18	AC	D0	00106		RET			
	04	A7		01	D0	00107	13\$:	MOVL	INPUT_ROUT, INPUT_ROUTINE		1004
				04	D0	0010B		MOVL	#1, PROMPT_FLAGS		1005
	04	A7		01	CE	0010F		BRB	15\$		0999
08		67		01	E0	00111	14\$:	MNEGL	#1, PROMPT_FLAGS		1007
04		67		01	E0	00115	15\$:	BBS	#1, HELP_FLAGS, 16\$		1009
06		67		02	E0	00119		BBS	#2, HELP_FLAGS, 16\$		1010
	01	A7		03	E1	0011D		BBC	#3, HELP_FLAGS, 17\$		
				08	88	00121	16\$:	BISB2	#8, HELP_FLAGS+1		1011
				04	11	00125		BRB	18\$		
	01	A7		08	8A	00127	17\$:	BICB2	#8, HELP_FLAGS+1		1012
21	01	A7		02	E1	0012B	18\$:	BBC	#2, HELP_FLAGS+1, 19\$		1014
				52	D4	00130		CLRL	NOMSG		1018
				52	DD	00132		PUSHL	NOMSG		1019
			10	AE	9F	00134		PUSHAB	LIBRARYNAME		
			011C	CE	9F	00137		PUSHAB	MAIN_LIBINDEX		
	0000V	CF		03	FB	0013B		CALLS	#3, OPEN_LIBRARY		
		76		50	E9	00140		BLBC	LOCAL_STATUS, 26\$		1020
	0118	CE		0114	CE	D0	00143	MOVL	MAIN_LIBINDEX, LAST_LIBINDEX		1021
	011C	CE		02	CE	0014A		MNEGL	#2, LAST_LIBNUMBER		1022
				3D	11	0014F		BRB	24\$		1014
19	01	A7		03	E1	00151	19\$:	BBC	#3, HELP_FLAGS+1, 21\$		1031
		6E		01	CE	00156		MNEGL	#1, LIBNO		1033
		52		01	7D	00159		MOVQ	#1, NOMSG		1035
		2B		53	E8	0015C	20\$:	BLBS	STATUS, 23\$		1036
				5E	DD	0015F		PUSHL	SP		1038
			08	AE	9F	00161		PUSHAB	ACMODE		
			14	AE	9F	00164		PUSHAB	LIBRARYNAME		
	0000V	CF		03	FB	00167		CALLS	#3, TRAN_NEXT_LIB		
		08		50	E8	0016C		BLBS	R0, 22\$		
		50	00000000G	8F	D0	0016F	21\$:	MOVL	#LBR\$_NOHLPLIBS, R0		1039
					04	00176		RET			
				52	DD	00177	22\$:	PUSHL	NOMSG		1040
			10	AE	9F	00179		PUSHAB	LIBRARYNAME		
			0120	CE	9F	0017C		PUSHAB	LAST_LIBINDEX		
	0000V	CF		03	FB	00180		CALLS	#3, OPEN_LIBRARY		
		53		50	D0	00185		MOVL	R0, STATUS		
				D2	11	00188		BRB	20\$		1036
			011C	CE	D4	0018A	23\$:	CLRL	LAST_LIBNUMBER		1042
			0C	AE	9F	0018E	24\$:	PUSHAB	LIBRARYNAME		1048
			0118	CE	9F	00191		PUSHAB	INDICES		
			0050	8F	BB	00195		PUSHR	#^M<R4,R6>		
			18	AE	9F	00199		PUSHAB	OUTPUT_WIDTH		
			0134	CE	9F	0019C		PUSHAB	GETCMD_DESC		
	0000V	CF		06	FB	001A0		CALLS	#6, PROMPT_HELP		
		53		50	D0	001A5		MOVL	R0, STATUS		
09	01	A7		02	E1	001A8		BBC	#2, HELP_FLAGS+1, 25\$		1051
				0114	CE	9F	001AD	PUSHAB	MAIN_LIBINDEX		1052
	0000V	CF		01	FB	001B1		CALLS	#1, CLOSE_LIBRARY		
		50		53	D0	001B6	25\$:	MOVL	STATUS, R0		1054
				04	001B9		26\$:	RET			1056

; Routine Size: 442 bytes, Routine Base: \$CODE\$ + 0000

```

332 1057 1 %SBITL 'Routine prompt_help';
333 1058 1 ROUTINE prompt_help (getcmd_desc, output_width, input_routine,
334 1059 1                               output_routine, indices, libraryname) =
335 1060 2 BEGIN
336 1061 2
337 1062 2 !++
338 1063 2 FUNCTIONAL DESCRIPTION:
339 1064 2
340 1065 2     This routine contains the interactive code loop that is repeatedly
341 1066 2     executed when help is being run in prompting mode. This same loop
342 1067 2     is executed exactly once when noprompt is specified.
343 1068 2
344 1069 2 INPUTS:
345 1070 2
346 1071 2     getcmd_desc =    address of the descriptor for the set of keys
347 1072 2                   to be processed
348 1073 2
349 1074 2     output_width =  address of longword containing width of output line
350 1075 2
351 1076 2     input_routine = address of user supplied input routine
352 1077 2
353 1078 2     output_routine = addresss of user supplied output routine
354 1079 2
355 1080 2     indices =        address of data structure containing indices of
356 1081 2                   libraries currently in use
357 1082 2
358 1083 2     libraryname =    address of string desc for default help library name
359 1084 2
360 1085 2 OUTPUTS:
361 1086 2
362 1087 2     None.
363 1088 2
364 1089 2 ROUTINE VALUE:
365 1090 2
366 1091 2     Always true.
367 1092 2
368 1093 2 --
369 1094 2 MAP
370 1095 2     getcmd_desc : REF BBLOCK,
371 1096 2     indices : REF BBLOCK,
372 1097 2     libraryname : REF BBLOCK;
373 1098 2
374 1099 2 LOCAL
375 1100 2     char_pos,
376 1101 2     key_descs : VECTOR [dsc$c_s_bln * hlp$c_maxkeys, BYTE],
377 1102 2     key_length_array : VECTOR [hlp$c_maxkeys],
378 1103 2     lib_name : BBLOCK [dsc$c_s_bln],
379 1104 2     lib_name_buf : VECTOR [filename_length, BYTE],
380 1105 2     print_data : BBLOCK [hpd$c_length],
381 1106 2     topic_prompt : BBLOCK [dsc$c_s_bln],
382 1107 2     topic_prompt_buf : BBLOCK [hlp$c_pagesize],
383 1108 2     sub_prompt : BBLOCK [dsc$c_s_bln],
384 1109 2     sub_prompt_level,
385 1110 2     sub_prompt_line : BBLOCK [hlp$c_pagesize],
386 1111 2     sub_prompt_buf : BBLOCK [hlp$c_pagesize],
387 1112 2     status;
388 1113 2
```

! Position of first non-blank character in input str
! String descriptors for keys
! Array of key lengths
! Library name descriptor
! Library name buffer
! Data structure for output driver
! Topic prompt descriptor
! Topic prompt buffer
! Sub-prompt descriptor
! Current key depth
! Sub-prompt line
! Sub-prompt buffer


```
389 1114 2 BIND
390 1115 2   main_libindex = indices [hli$l_mainindex],      ! Index of /LIB library
391 1116 2   last_libindex = indices [hli$l_lastindex],      ! Index of last library examined
392 1117 2   last_libnumber = indices [hli$l_lastnumb],      ! No. of last lib examined, relative to all default
393 1118 2   getcmd_line = .getcmd_desc [dsc$a_pointer],      ! Command buffer
394 1119 2   true_keys = print_data [hpd$b_truekeys] : SIGNED BYTE, ! Number of help keys
395 1120 2   help_level = print_data [hpd$b_helplevel] : BYTE, ! Current key depth
396 1121 2   print_flags = print_data [hpd$b_printflag] : BLOCK, ! Flags for output driver
397 1122 2   add_info_level = print_data [hpd$l_subpmtlev], ! Level of additional info
398 1123 2   sub_prompt_ptr = print_data [hpd$l_subpmtptr], ! Ptr used for filling sub-prompt buffer
399 1124 2   length_array = print_data [hpd$l_lenarray] : REF VECTOR, ! Address of key length array
400 1125 2   outputroutine = print_data [hpd$l_outputrou]; ! User specified output routine
401 1126 2
402 1127 2   length_array = key_length_array; ! Init print_data array
403 1128 2   outputroutine = .output_routine;
404 1129 2
405 1130 2   lib_name = 0; ! Initialize library name
406 1131 2   lib_name [dsc$a_pointer] = lib_name_buf;
407 1132 2   IF NOT .help_flags [hlp$v_library] ! If no main library
408 1133 2   THEN switch_libname (.libraryname, lib_name); ! Then use passed library name in pr
409 1134 2
410 1135 2   topic_prompt = 0; ! Initialize topic prompt
411 1136 2   topic_prompt [dsc$a_pointer] = topic_prompt_buf;
412 1137 2   CH$MOVE (.prompt_prefix [0], prompt_prefix [1], topic_prompt_buf);
413 1138 2
414 1139 2   sub_prompt = 0; ! Initialize sub-prompt
415 1140 2   sub_prompt [dsc$a_pointer] = sub_prompt_line;
416 1141 2   CH$MOVE (.prompt_prefix [0], prompt_prefix [1], sub_prompt_line);
417 1142 2
418 1143 3   WHILE (.getcmd_desc [dsc$w_length] GTR 0) ! Remove any preceeding blanks from
419 1144 3   AND (CH$RCHAR (.getcmd_desc [dsc$a_pointer]) EQL %C' ')
420 1145 3   DO BEGIN
421 1146 3   getcmd_desc [dsc$w_length] = .getcmd_desc [dsc$w_length] - 1;
422 1147 3   getcmd_desc [dsc$a_pointer] = .getcmd_desc [dsc$a_pointer] + 1;
423 1148 3   END;
424 1149 2
425 1150 3   IF (.getcmd_desc [dsc$w_length] GTR 0) ! If non-empty help keys
426 1151 3   AND (CH$RCHAR (.getcmd_desc [dsc$a_pointer]) EQL %C'a') ! And initial key starts with 'a'
427 1152 3   THEN (change_lib (.getcmd_desc, .getcmd_desc [dsc$a_pointer], ! Then change to specified library
428 1153 3   .indices, lib_name));
429 1154 2
430 1155 2   WHILE (.prompt_flags AND hcf$m_cont) NEQ 0 ! WHILE we can still continue...
431 1156 3   DO BEGIN
432 1157 3
433 1158 3   IF .prompt_flags EQL hcf$m_noprompt ! If one shot command
434 1159 3   THEN prompt_flags = NOT hcf$m_noprompt; ! Execute while exactly once
435 1160 3
436 1161 3   IF .prompt_flags GTR hcf$m_stay ! If prompting enabled
437 1162 4   THEN BEGIN ! Prompt for keywords
438 1163 4
439 1164 4   !
440 1165 4   ! If moving down a prompt level then update the subtopic prompt.
441 1166 4   !
442 1167 4
443 1168 4   IF (.prompt_flags AND (hcf$m_more OR hcf$m_stay)) EQL hcf$m_more
444 1169 5   THEN BEGIN
445 1170 5
```

```

446      1171 5      LOCAL
447      1172 5      sub_length;
448      1173 5
449      1174 5      IF .lib_name [dsc$w_length] NEQ 0      ! If not main library
450      1175 6      THEN BEGIN      ! Then insert library name in prompt
451      1176 6          CH$WCHAR (%C'a', sub_prompt_line + 4);
452      1177 6          CH$MOVE (.lib_name [dsc$w_length],
453      1178 6              .lib_name [dsc$a_pointer],
454      1179 6              sub_prompt_line + 4 + 1);
455      1180 6          CH$WCHAR (%X'20', sub_prompt_line + 4 + 1 + .lib_name [dsc$w_length]);
456      1181 6          sub_prompt [dsc$w_length] = 4 + 1 + .lib_name [dsc$w_length] + 1;
457      1182 6      END
458      1183 5      ELSE sub_prompt [dsc$w_length] = 4;      ! Otherwise skip to keys
459      1184 5
460      1185 5      sub_length = .sub_prompt_ptr - sub_prompt_buf;      ! Calculate length of keys
461      1186 5      CH$MOVE (.sub_length, sub_prompt_buf,      ! Move keys into prompt line
462      1187 5          sub_prompt_line + .sub_prompt [dsc$w_length]);
463      1188 5      CH$MOVE (.subtopic [0], subtopic [1],      ! Move 'subtopic?' into prompt line
464      1189 5          sub_prompt_line + .sub_prompt [dsc$w_length] + .sub_length);
465      1190 5      sub_prompt [dsc$w_length] = .sub_prompt [dsc$w_length]      ! Update prompt length
466      1191 5          + .sub_length + .subtopic [0];
467      1192 5      sub_prompt_level = .help_level;      ! Update prompt level
468      1193 4      END;
469      1194 4
470      1195 4      !
471      1196 4      ! If subprompt length greater than zero, then do subtopic prompting.
472      1197 4      !
473      1198 4
474      1199 4      IF .sub_prompt [dsc$w_length] NEQ 0
475      1200 5      THEN BEGIN
476      1201 5          LOCAL
477      1202 5          prefix_length,
478      1203 5          sub_length;
479      1204 5
480      1205 5
481      1206 5      IF .lib_name [dsc$w_length] NEQ 0      ! If not main library
482      1207 5      THEN prefix_length = 4 + 1 + .lib_name [dsc$w_length] + 1      ! Then prefix includes lib name
483      1208 5      ELSE prefix_length = 4;      ! Else it doesn't
484      1209 5
485      1210 5      sub_length = .sub_prompt [dsc$w_length] - .prefix_length      ! Calculate length of keys
486      1211 5          - .subtopic [0];
487      1212 5      CH$MOVE (.sub_length, sub_prompt_line + .prefix_length,      ! Move keys into command line
488      1213 5          getcmd_line);
489      1214 5      getcmd_desc [dsc$w_length] = hlp$c_pagesize - .sub_length;      ! Calculate space left in command li
490      1215 5      getcmd_desc [dsc$a_pointer] = getcmd_line + .sub_length;      ! Set pointer to end of keys
491      1216 5
492      1217 6      IF (status = (.input_routine) (getcmd_desc [dsc$w_length],
493      1218 5          sub_prompt, getcmd_desc [dsc$w_length])) EQL RMS$_EOF      ! Get input
494      1219 5      THEN EXITLOOP      ! If CNTL-Z
495      1220 5      ELSE IF NOT .status      ! Then get out of help
496      1221 5      THEN RETURN lbr$_usrinperr;      ! Else if other error
497      1222 5      ! Then signal user input error
498      1223 7
499      1224 6      IF ((char_pos = CH$FIND_NOT_CH (.getcmd_desc [dsc$w_length],
500      1225 6          .getcmd_desc [dsc$a_pointer], %ASCII ' ')) EQL 0)      ! If blank line
501      1226 6      OR (CH$RCHAR (.char_pos) EQL %X'0D')      ! Or CR
502      1227 5      THEN IF (sub_prompt_level = .sub_prompt_level - 1) EQL 0      ! Then back up a prompt level
      ! If now at topic level
```



```

503      1228 5      THEN sub_prompt [dsc$w_length] = 0      ! then clear subtopic prompt
504      1229 6      ELSE BEGIN
505      1230 6          remove_last_key (sub_prompt,      ! else remove last key
506      1231 6              .key_length_array [.sub_prompt_level+1]);
507      1232 6          prompt_flags = hcf$m_backup OR hcf$m_cont;      ! Set backup flag
508      1233 6          END
509      1234 6
510      1235 6      ELSE SELECTONE (CH$RCHAR (.getcmd_desc [dsc$w_length] +      ! Test termination character
511      1236 5          .getcmd_desc [dsc$a_pointer] - 1) OF SET
512      1237 5
513      1238 5          [%C'?'']:      ! If ?
514      1239 6          BEGIN      ! then repeat help for old help keys
515      1240 6              getcmd_desc [dsc$w_length] = .sub_length;
516      1241 6              getcmd_desc [dsc$a_pointer] = getcmd_line;
517      1242 5          END;
518      1243 5
519      1244 5          [%X'1A']:      ! If CNTL-Z
520      1245 5          EXITLOOP;      ! then get out of help
521      1246 5
522      1247 5          [OTHERWISE]:      ! If help keys
523      1248 6          BEGIN      ! then append command to old keys
524      1249 6              getcmd_desc [dsc$w_length] = .getcmd_desc [dsc$w_length] + .sub_length;
525      1250 6              getcmd_desc [dsc$a_pointer] = getcmd_line;
526      1251 5          END;
527      1252 5          TES;
528      1253 5
529      1254 5          remove_terminator (.getcmd_desc);      ! Remove termination character from
530      1255 5
531      1256 4          END;
532      1257 4
533      1258 4      ! If subtopic length equals zero, then do topic prompting
534      1259 4      !
535      1260 4      !
536      1261 4      !
537      1262 4      IF .sub_prompt [dsc$w_length] EQL 0
538      1263 5      THEN BEGIN
539      1264 5
540      1265 5          getcmd_desc [dsc$w_length] = hlp$c_pagesize;      ! Init rest of descriptor
541      1266 5          getcmd_desc [dsc$a_pointer] = getcmd_line;
542      1267 5
543      1268 5          IF .last_libnumber GEQ external_libnumber      ! If not main library
544      1269 6          THEN BEGIN      ! Then insert library name in prompt
545      1270 6              CH$WCHAR (%C'a', topic_prompt_buf + 4);
546      1271 6              CH$MOVE (.lib_name [dsc$w_length],
547      1272 6                  .lib_name [dsc$a_pointer],
548      1273 6                  topic_prompt_buf + 4 + 1);
549      1274 6              CH$WCHAR (%X'20', topic_prompt_buf + 4 + 1 +
550      1275 6                  .lib_name [dsc$w_length]);
551      1276 6              CH$MOVE (.topic [0], topic [1],
552      1277 6                  topic_prompt_buf + 4 + 1 + .lib_name [dsc$w_length] + 1);
553      1278 6              topic_prompt [dsc$w_length] = 4 + 1 +
554      1279 6                  .lib_name [dsc$w_length] + 1 + .topic [0];
555      1280 6          END
556      1281 6      ELSE BEGIN      ! Otherwise, do not include library
557      1282 6          topic_prompt [dsc$w_length] = 4 + .topic [0];
558      1283 6          CH$MOVE (.topic [0], topic [1],
559      1284 6              topic_prompt_buf + 4);
```

```
560      1285 5      END;
561      1286 5
562      1287 6      IF (status = (.input_routine) (getcmd_desc [dsc$w_length],
563      1288 5      topic_prompt, getcmd_desc [dsc$w_length])) EQL RMSS_EOF
564      1289 5      THEN EXITLOOP
565      1290 5      ELSE IF NOT .status
566      1291 5      THEN RETURN lbr$_usrinperr;
567      1292 5
568      1293 7      IF ((char_pos = CH$FIND_NOT_CH (.getcmd_desc [dsc$w_length],
569      1294 6      .getcmd_desc [dsc$a_pointer], %ASCII ' ')) EQL 0)
570      1295 6      OR (CH$RCHAR (.char_pos) EQL %X'0D')
571      1296 7      THEN (IF (.last_libnumber LSS external_libnumber
572      1297 7      OR NOT .help_flags [hlp$v_library])
573      1298 6      THEN EXITLOOP
574      1299 7      ELSE BEGIN
575      1300 7      close_library (last_libindex);
576      1301 7      last_libnumber = main_libnumber;
577      1302 7      last_libindex = .main_libindex;
578      1303 7      lib_name [dsc$w_length] = 0;
579      1304 7      getcmd_desc [dsc$w_length] = 0;
580      1305 7      getcmd_desc [dsc$a_pointer] = getcmd_line;
581      1306 7      prompt_flags = hcf$m_backup OR hcf$m_cont;
582      1307 6      END;)
583      1308 6
584      1309 6      ELSE SELECTONE (CH$RCHAR (.getcmd_desc [dsc$w_length] +
585      1310 5      .getcmd_desc [dsc$a_pointer] - 1)) OF SET
586      1311 5
587      1312 5      [XC'?' ]:
588      1313 6      BEGIN
589      1314 6      getcmd_desc [dsc$w_length] = 0;
590      1315 6      getcmd_desc [dsc$a_pointer] = getcmd_line;
591      1316 5      END;
592      1317 5
593      1318 5      [XX'1A' ]:
594      1319 5      EXITLOOP;
595      1320 5
596      1321 5      [OTHERWISE]:
597      1322 6      IF (CH$RCHAR (.char_pos) EQL %C'a')
598      1323 6      THEN (change_lib (.getcmd_desc, .char_pos,
599      1324 5      .indices, lib_name));
600      1325 5      TES;
601      1326 5
602      1327 5      remove_terminator (.getcmd_desc);
603      1328 5
604      1329 4      END;
605      1330 4
606      1331 3      END;
607      1332 3
608      1333 3      IF (.prompt_flags AND hcf$m_backup) EQL 0
609      1334 4      THEN BEGIN
610      1335 5      IF .prompt_flags NEQ (NOT hcf$m_noprompt)
611      1336 4      THEN prompt_flags = hcf$m_cont;
612      1337 4      sub_prompt_ptr = sub_prompt_buf;
613      1338 4      help_level = 0;
614      1339 4      add_info_level = 0;
615      1340 4      setup_keys (.getcmd_desc, key_descs, true_keys);
616      1341 4      print_flags [hpd$v_init] = 0;
```

```
! Get input
! If CNTL-Z
! Then get out of help
! Else if other error
! Then signal user input error

! If blank line
! Or CR
! Then back up a library
! If already at main library
! or if there is no main library
! Then get out of help
! Else back up to main library
! Close old user library
! Set main library number
! And index
! Reset library name
! Clear command descriptor

! Set back up flag

! Test termination character

! If ?
! Then repeat help for old library

! If CNTL-Z
! Then get out of help

! If help keys
! And start with 'a'

! Then change to specified library

! Remove termination character from

! Of prompt for keywords

! If not backing up a level
! then get help text
! If not prompting
! Say to continue prompting
! Init subtopic buffer pointer
! Set current key depth to zero

! Set up individual help key desc's
! Set print not initialized flag
```



```

: 617      1342  4      end topic_flag = false;
: 618      1343  5      IF NOT (status = search_libs (key_descs, .output_width, print_data,
: 619      1344  5          .indices, lib_name, .sub_prompt [desc$w_length]))
: 620      1345  4          THEN RETURN .status;
: 621      1346  4      IF .print_flags [hpd$w_init]
: 622      1347  4          THEN nohelp_log (.getcmd_desc);
: 623      1348  4      IF .help_flags [hlp$w_liblist] AND
: 624      1349  4          NOT .end_topic_flag AND
: 625      1350  5          ((.true_keys LEQ 0) OR
: 626      1351  5          ((.add_info_level EQL 0) AND .print_flags [hpd$w_init]))
: 627      1352  4          THEN
: 628      1353  4              libs_available (.output_routine, .output_width);
: 629      1354  4      END
: 630      1355  3  ELSE prompt_flags = hcf$m_cont OR hcf$m_stay;
: 631      1356  3
: 632      1357  2  END;
: 633      1358  2
: 634      1359  2  RETURN true
: 635      1360  2
: 636      1361  1  END;

```

```

                                OFFC 00000 PROMPT_HELP:
                                .WORD
                                Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
                                : 1058
                                5E      F92C  CE  9E 00002      MOVAB      -1748(SP), SP
                                5A      14  AC  D0 00007      MOVL      INDICES, R10
                                58      04  AC  D0 0000B      MOVL      GETCMD_DESC, R8
                                59      04  A8  9E 0000F      MOVAB      4(R8), R9
                                69      DD 00013      PUSHL     (R9)
                                FF50  CD   88  AD 9E 00015      MOVAB      KEY_LENGTH_ARRAY, LENGTH_ARRAY
                                FF4C  CD   10  AC  D0 0001B      MOVL      OUTPUT_ROUTINE, OUTPUTROUTINE
                                80      AD  D4 00021      CLRL      LIB_NAME
                                84      AD  FF58 CD 9E 00024      MOVAB      LIB_NAME_BUF, LIB_NAME+4
                                OB      0000' CF   02  E0 0002A      BBS       #2, HELP_FLAGS+1, -1$
                                80      AD  9F 00030      PUSHAB    LIB_NAME
                                18      AC  DD 00033      PUSHL     LIBRARYNAME
                                0000V  CF   02  FB 00036      CALLS     #2, SWITCH_LIBNAME
                                FF3C  CD  D4 0003B  1$:      CLRL      TOPIC_PROMPT
                                FF40  CD  FD3C CD 9E 0003F      MOVAB      TOPIC_PROMPT_BUF, TOPIC_PROMPT+4
                                50      0000' CF  9A 00046      MOVZBL    PROMPT_PREFIX, R0
                                FD3C  CD  50  28 0004B      MOVCL     R0, PROMPT_PREFIX+1, TOPIC_PROMPT_BUF
                                FD38  CD  FD34 CD  D4 00053      CLRL      SUB_PROMPT
                                020C  CE  020C CE  9E 00057      MOVAB      SUB_PROMPT_LINE, SUB_PROMPT+4
                                50      0000' CF  9A 0005E      MOVZBL    PROMPT_PREFIX, R0
                                020C  CE  0000' CF  50  28 00063      MOVCL     R0, PROMPT_PREFIX+1, SUB_PROMPT_LINE
                                68      B5 0006B  2$:      TSTW      (R8)
                                0C      13 0006D      BEQL      3$
                                20      00  B9  91 0006F      CMPB      @0(R9), #32
                                06      12 00073      BNEQ      3$
                                68      B7 00075      DECB      (R8)
                                69      D6 00077      INCL      (R9)
                                F0      11 00079      BRB       2$
                                68      B5 0007B  3$:      TSTW      (R8)
                                15      13 0007D      BEQL      4$
                                : 1143
                                : 1144
                                : 1146
                                : 1147
                                : 1143
                                : 1150
                                :

```

40	8F	00	B9	91	0007F	CMPB	@0(R9), #64	1151			
			0E	12	00084	BNEQ	4\$				
		80	AD	9F	00086	PUSHAB	LIB_NAME	1152			
			5A	DD	00089	PUSHL	R10	1153			
			69	DD	0008B	PUSHL	(R9)	1152			
			58	DD	0008D	PUSHL	R8				
0000V	CF		04	FB	0008F	CALLS	#4, CHANGE_LIB				
	03	0000'	CF	E8	00094	BLBS	PROMPT_FLAGS, 5\$	1155			
			02D4	31	00099	BRW	45\$				
FFFFFFFF	8F	0000'	CF	D1	0009C	CMPL	PROMPT_FLAGS, #-1	1158			
			04	12	000A5	BNEQ	6\$				
		0000'	CF	D4	000A7	CLRL	PROMPT_FLAGS	1159			
	02	0000'	CF	D1	000AB	CMPL	PROMPT_FLAGS, #2	1161			
			03	14	000B0	BGTR	7\$				
			021F	31	000B2	BRW	37\$				
50	0000'	CF	FFFFF9	8F	CB	000B5	BICL3	#-7, PROMPT_FLAGS, R0	1168		
		04		50	D1	000BF	CMPL	R0, #4			
				5E	12	000C2	BNEQ	10\$			
		56	80	AD	3C	000C4	MOVZWL	LIB_NAME, R6	1174		
				1B	13	000C8	BEQL	8\$			
	0210	CE	40	8F	90	000CA	MOVB	#64, SUB_PROMPT_LINE+4	1176		
0211	CE	84	BD	56	28	000D0	MOVC3	R6, @LIB_NAME+4, SUB_PROMPT_LINE+5	1179		
FD34	CD	0211	CE46	20	90	000D7	MOVB	#32, SUB_PROMPT_LINE+5[R6]	1180		
			56	06	A1	000DD	ADDW3	#6, R6, SUB_PROMPT	1181		
				05	11	000E3	BRB	9\$	1174		
	FD34	CD		04	B0	000E5	MOVW	#4, SUB_PROMPT	1183		
		50	0C	AE	9E	000EA	MOVAB	SUB_PROMPT_BUF, R0	1185		
57	FF44	CD		50	C3	000EE	SUBL3	R0, SUB_PROMPT_PTR, SUB_LENGTH			
		56	FD34	CD	3C	000F4	MOVZWL	SUB_PROMPT, R6	1187		
020C	CE46	0C	AE	57	28	000F9	MOVC3	SUB_LENGTH, SUB_PROMPT_BUF, SUB_PROMPT_LINE-[R6]			
			50	0000'	CF	9A	00101	MOVZBL	SUBTOPIC, R0	1188	
			56		57	C0	00106	ADDL2	SUB_LENGTH, R6	1189	
020C	CE46	0000'	CF	50	28	00109	MOVC3	R0, SUBTOPIC+1, SUB_PROMPT_LINE[R6]			
			50	0000'	CF	9A	00112	MOVZBL	SUBTOPIC, R0	1191	
FD34	CD		56	50	A1	00117	ADDW3	R0, R6, SUB_PROMPT			
			58	FF55	CD	9A	0011D	MOVZBL	HELP_LEVEL, SUB_PROMPT_LEVEL	1192	
			51	FD34	CD	3C	00122	MOVZWL	SUB_PROMPT, R1	1199	
					03	12	00127	BNEQ	11\$		
				00A9	31	00129	BRW	23\$			
			80	AD	B5	0012C	TSTW	LIB_NAME	1206		
				09	13	0012F	BEQL	12\$			
		50	80	AD	3C	00131	MOVZWL	LIB_NAME, PREFIX_LENGTH	1207		
		50		06	C0	00135	ADDL2	#6, PREFIX_LENGTH			
				03	11	00138	BRB	13\$			
		50		04	D0	0013A	MOVL	#4, PREFIX_LENGTH	1208		
		51		50	C2	0013D	SUBL2	PREFIX_LENGTH, R1	1210		
		56	0000'	CF	9A	00140	MOVZBL	SUBTOPIC, SUB_LENGTH	1211		
		51		56	C3	00145	SUBL3	SUB_LENGTH, RT, SUB_LENGTH			
00	56	020C	CE40	56	28	00149	MOVC3	SUB_LENGTH, SUB_PROMPT_LINE-[PREFIX_LENGTH], @0(SP)	1212		
	68	0200	8F	56	A3	00151	SUBW3	SUB_LENGTH, #512, (R8)	1214		
	69		6E	56	C1	00157	ADDL3	SUB_LENGTH, (SP), (R9)	1215		
				58	DD	0015B	PUSHL	R8	1218		
				FD34	CD	9F	0015D	PUSHAB	SUB_PROMPT	1217	
					58	DD	00161	PUSHL	R8	1218	
		0C	BC	03	FB	00163	CALLS	#3, @INPUT_ROUTINE			

00	B9	04	AE	50	D0	00167	MOVL	R0, STATUS	:
		0001827A	8F	04	AE	D1 00168	CMPL	STATUS, #98938	:
			03	04	AE	13 00173	BEQL	20\$:
			68	00D1	E8 00175	BLBS	STATUS, 14\$:	1220
					31 00179	BRW	28\$:	1223
				20	3B 0017C	SKPC	#32, (R8), @0(R9)	:	
				02	12 00181	BNEQ	15\$:	
				51	D4 00183	CLRL	R1	:	
		08	AE	51	D3 00185	MOVL	R1, CHAR_POS	:	
			0D	06	13 00189	BEQL	16\$:	1224
				08	BE 91 0018B	CMPB	@CHAR_POS, #13	:	1225
				1E	12 0018F	BNEQ	18\$:	
				5B	D7 00191	DECL	SUB_PROMPT_LEVEL	:	1227
				06	12 00193	BNEQ	17\$:	
				FD34	CD B4 00195	CLRW	SUB_PROMPT	:	1228
					33 11 00199	BRB	22\$:	
				8C	AD4B DD 0019B	PUSHL	KEY_LENGTH_ARRAY+4[SUB_PROMPT_LEVEL]	:	1231
				FD34	CD 9F 0019F	PUSHAB	SUB_PROMPT	:	1230
		0000V	CF	02	FB 001A3	CALLS	#2, REMOVE_LAST_KEY	:	
		0000'	CF	11	D0 001A8	MOVL	#17, PROMPT_FLAGS	:	1232
				1F	11 001AD	BRB	22\$:	1227
			50	68	3C 001AF	MOVZWL	(R8), R0	:	1236
			50	69	C0 001B2	ADDL2	(R9), R0	:	
			50	FF	A0 9A 001B5	MOVZBL	-1(R0), R0	:	
			3F	50	91 001B9	CMPB	R0, #63	:	1238
				05	12 001BC	BNEQ	19\$:	
			68	56	B0 001BE	MOVW	SUB_LENGTH, (R8)	:	1240
				08	11 001C1	BRB	21\$:	1241
			1A	50	91 001C3	CMPB	R0, #26	:	1244
				7F	13 001C6	BEQL	27\$:	
			68	56	A0 001C8	ADDW2	SUB_LENGTH, (R8)	:	1249
			69	6E	D0 001CB	MOVL	(SPT), (R9)	:	1250
				58	DD 001CE	PUSHL	R8	:	1254
		0000V	CF	01	FB 001D0	CALLS	#1, REMOVE_TERMINATOR	:	
				FD34	CD B5 001D5	TSTW	SUB_PROMPT	:	1262
					03 13 001D9	BEQL	24\$:	
				00F6	31 001DB	BRW	37\$:	
			68	0200	8F B0 001DE	MOVW	#512, (R8)	:	1265
			69		6E D0 001E3	MOVL	(SP), (R9)	:	1266
			57	0000'	CF 9A 001E6	MOVZBL	TOPIC, R7	:	1276
		FFFFFFFF	8F	08	AA D1 001EB	CMPL	8(R10), #-1	:	1268
				2C	19 001F3	BLSS	25\$:	
		FD40	CD	40	8F 90 001F5	MOVB	#64, TOPIC_PROMPT_BUF+4	:	1270
			56	80	AD 3C 001FB	MOVZWL	LIB_NAME, R6	:	1271
FD41	CD	84	BD	56	28 001FF	MOV3	R6, @LIB_NAME+4, TOPIC_PROMPT_BUF+5	:	1273
		FD41	CD46	20	90 00206	MOVB	#32, TOPIC_PROMPT_BUF+5[R6]	:	1275
FD42	CD46	0000'	CF	57	28 0020C	MOV3	R7, TOPIC+T, TOPIC_PROMPT_BUF+6[R6]	:	1277
			50	06	A746 9E 00215	MOVAB	6(R7)[R6], R0	:	1279
		FF3C	CD	50	B0 0021A	MOVW	R0, TOPIC_PROMPT	:	
				0E	11 0021F	BRB	26\$:	1268
			57	04	A1 00221	ADDW3	#4, R7, TOPIC_PROMPT	:	1282
FF3C	CD	0000'	CF	57	28 00227	MOV3	R7, TOPIC+1, TOPIC_PROMPT_BUF+4	:	1284
FD40	CD			58	DD 0022F	PUSHL	R8	:	1288
				FF3C	CD 9F 00231	PUSHAB	TOPIC_PROMPT	:	1287
					58 DD 00235	PUSHL	R8	:	1288
		0C	BC	03	FB 00237	CALLS	#3, @INPUT_ROUTINE	:	
		04	AE	50	D0 0023B	MOVL	R0, STATUS	:	

0001827A	8F	04	AE	D1	0023F	CMPL	STATUS, #98938	:	
	08	04	2B	13	00247	BEQL	32\$:	1290
	50	00000000G	AE	E8	00249	BLBS	STATUS, 29\$:	1291
			8F	D0	0024D	MOVL	#LBR\$_USRINPERR, R0	:	
00	B9			04	00254	RET		:	1293
	68		20	3B	00255	SKPC	#32, (R8), a0(R9)	:	
			02	12	0025A	BNEQ	30\$:	
	08	AE	51	D4	0025C	CLRL	R1	:	
			51	D0	0025E	MOVL	R1, CHAR_POS	:	1294
	0D	08	06	13	00262	BEQL	31\$:	1295
			BE	91	00264	CMPB	@CHAR_POS, #13	:	
	FFFFFFF	8F	32	12	00268	BNEQ	34\$:	1296
			AA	D1	0026A	CMPL	8(R10), #-1	:	
			03	18	00272	BGEQ	33\$:	
F7	0000'	CF	00F9	31	00274	BRW	45\$:	1297
			02	E1	00277	BBC	#2, HELP_FLAGS+1, 32\$:	1300
	0000V	CF	04	AA	9F	PUSHAB	4(R10)	:	
	08	AA	01	FB	00280	CALLS	#1, CLOSE_LIBRARY	:	1301
	04	AA	02	CE	00285	MNEGL	#2, 8(R10)	:	1302
			6A	D0	00289	MOVL	(R10), 4(R10)	:	1303
			80	AD	B4	CLRW	LIB_NAME	:	1304
			68	B4	00290	CLRW	(R8)	:	1305
	0000'	69	6E	D0	00292	MOVL	(SP), (R9)	:	1306
		CF	11	D0	00295	MOVL	#17, PROMPT_FLAGS	:	1293
			31	11	0029A	BRB	36\$:	1310
	50		68	3C	0029C	MOVZWL	(R8), R0	:	
	50		69	C0	0029F	ADDL2	(R9), R0	:	
	50		FF	A0	9A	MOVZBL	-1(R0), R0	:	
	3F		50	91	002A6	CMPB	R0, #63	:	1312
			07	12	002A9	BNEQ	35\$:	
			68	B4	002AB	CLRW	(R8)	:	1314
	69		6E	D0	002AD	MOVL	(SP), (R9)	:	1315
			1B	11	002B0	BRB	36\$:	1309
	1A		50	91	002B2	CMPB	R0, #26	:	1318
			BD	13	002B5	BEQL	32\$:	
	40	8F	08	BE	91	CMPB	@CHAR_POS, #64	:	1322
			0F	12	002BC	BNEQ	36\$:	
			80	AD	9F	PUSHAB	LIB_NAME	:	1323
			5A	DD	002C1	PUSHL	R10	:	1324
			10	AE	DD	PUSHL	CHAR_POS	:	1323
			58	DD	002C6	PUSHL	R8	:	
	0000V	CF	04	FB	002C8	CALLS	#4, CHANGE_LIB	:	1327
			58	DD	002CD	PUSHL	R8	:	
03	0000V	CF	01	FB	002CF	CALLS	#1, REMOVE_TERMINATOR	:	1333
	0000'	CF	04	E1	002D4	BBC	#4, PROMPT_FLAGS, 38\$:	
			008B	31	002DA	BRW	43\$:	
			0000'	CF	D5	TSTL	PROMPT_FLAGS	:	1335
			05	13	002E1	BEQL	39\$:	
	0000'	CF	01	D0	002E3	MOVL	#1, PROMPT_FLAGS	:	1336
	FF44	CD	0C	AE	9E	MOVAB	SUB_PROMPT_BUF, SUB_PROMPT_PTR	:	1337
			FF55	CD	94	CLRB	HELP_LEVEL	:	1338
			FF48	CD	D4	CLRL	ADD_INFO_LEVEL	:	1339
			FF54	CD	9F	PUSHAB	TRUE_KEYS	:	1340
			B0	AD	9F	PUSHAB	KEY_DESCS	:	
				58	DD	PUSHL	R8	:	
	0000V	CF	03	FB	002FF	CALLS	#3, SETUP_KEYS	:	
	FF56	CD	01	8A	00304	BICB2	#1, PRINT_FLAGS	:	1341

		0000'	CF	94	00309	CLRB	END_TOPIC_FLAG	:	1342
	7E	FD34	CD	3C	0030D	MOVZWL	SUB_PROMPT, -(SP)	:	1344
		80	AD	9F	00312	PUSHAB	LIB_NAME	:	1343
			5A	DD	00315	PUSHL	R10	:	1344
		FF44	CD	9F	00317	PUSHAB	PRINT_DATA	:	1343
		08	AC	DD	0031B	PUSHL	OUTPUT_WIDTH	:	
		80	AD	9F	0031E	PUSHAB	KEY_DESCS	:	
	0000V		06	FB	00321	CALLS	#6, SEARCH_LIBS	:	
	04		50	D0	00326	MOVL	R0, STATUS	:	
	05	04	AE	E8	0032A	BLBS	STATUS, 40\$:	
	50	04	AE	D0	0032E	MOVL	STATUS, R0	:	1345
			04	00332	RET			:	
	07	FF56	CD	E9	00333	BLBC	PRINT_FLAGS, 41\$:	1346
			58	DD	00338	PUSHL	R8	:	1347
	0000V		01	FB	0033A	CALLS	#1, NOHELP_LOG	:	
28	0000'		04	E1	0033F	BBC	#4, HELP_FLAGS, 44\$:	1348
			CF	E8	00345	BLBS	END_TOPIC_FLAG, 44\$:	1349
	23	0000'	CF	E8	00345	BLBS	END_TOPIC_FLAG, 44\$:	1349
		FF54	CD	95	0034A	TSTB	TRUE_KEYS	:	1350
			0B	15	0034E	BLEQ	42\$:	
		FF48	CD	D5	00350	TSTL	ADD_INFO_LEVEL	:	1351
			17	12	00354	BNEQ	44\$:	
	12	FF56	CD	E9	00356	BLBC	PRINT_FLAGS, 44\$:	
		08	AC	DD	0035B	PUSHL	OUTPUT_WIDTH	:	1353
		10	AC	DD	0035E	PUSHL	OUTPUT_ROUTINE	:	
	0000V		02	FB	00361	CALLS	#2, LIBS_AVAILABLE	:	
			05	11	00366	BRB	44\$:	1333
	0000'		03	D0	00368	MOVL	#3, PROMPT_FLAGS	:	1355
		FD24	31	0036D	BRW	4\$:	1155
	50		01	D0	00370	MOVL	#1, R0	:	1359
			04	00373	RET			:	1361

; Routine Size: 884 bytes, Routine Base: \$CODE\$ + 01BA

```

638 1362 1 %SBTTL 'Routine search_libs';
639 1363 1 ROUTINE search_libs (keydescs, outputwidth, printdata,
640 1364 1 indices, libname, subpromptlen) =
641 1365 2 BEGIN
642 1366 2
643 1367 2 !++
644 1368 2 FUNCTIONAL DESCRIPTION:
645 1369 2
646 1370 2 This routine searches the default help libraries and calls the
647 1371 2 librarian function to extract help from each help library as
648 1372 2 required.
649 1373 2
650 1374 2 INPUTS:
651 1375 2
652 1376 2 keydescs = address of vector of key descriptors
653 1377 2
654 1378 2 outputwidth = address of longword containing width of output line
655 1379 2
656 1380 2 printdata = address of data structure containing info for
657 1381 2 the output driver
658 1382 2
659 1383 2 indices = address of data structure containing indices of
660 1384 2 libraries currently in use
661 1385 2
662 1386 2 libname = address of descriptor for the user default library name
663 1387 2
664 1388 2 subpromptlen = total length of help keys, zero => at topic prompt level
665 1389 2
666 1390 2 OUTPUTS:
667 1391 2
668 1392 2 printdata = flags are manipulated by this routine and other
669 1393 2 values are altered by the output driver
670 1394 2
671 1395 2 indices = updated to reflect library that help information
672 1396 2 was eventually extracted from
673 1397 2
674 1398 2 libname = if help found in default user library, updated to
675 1399 2 the file name of that library
676 1400 2
677 1401 2 ROUTINE VALUE:
678 1402 2
679 1403 2 Always true.
680 1404 2
681 1405 2 --
682 1406 2 MAP
683 1407 2 indices : REF BBLOCK,
684 1408 2 keydescs : REF BBLOCK,
685 1409 2 libname : REF BBLOCK,
686 1410 2 printdata : REF BBLOCK;
687 1411 2
688 1412 2 BIND
689 1413 2 main_libindex = indices [hli$L_mainindex], ! Index of /LIB library
690 1414 2 last_libindex = indices [hli$L_lastindex], ! Index of last library examined
691 1415 2 last_libnumber = indices [hli$L_lastnumb], ! No. of last library examined, relative to all defa
692 1416 2 print_flags = printdata [hpd$b_printflag] : BBLOCK; ! Flags for output driver
693 1417 2
694 1418 2 LOCAL
```



```
: 695      1419 2      current_libindex,      ! Index of library currently being searched
: 696      1420 2      current_libnumber,      ! Number of library currently being searched
: 697      1421 2      deflib_acmode : BYTE,      ! Logical name table number
: 698      1422 2      librarystring : BBLOCK [nam$c_maxrss],      ! Default library name string
: 699      1423 2      libraryname : BBLOCK [dsc$c_s_bln],      ! String descriptor for library name
: 700      1424 2      nomsg,      ! Open library message flag
: 701      1425 2      user_libno,      ! HLP$LIBRARY number
: 702      1426 2      status;
: 703      1427 2
: 704      1428 2      nomsg = true;      ! Do not signal error if library can't be opened
: 705      1429 2      print_flags [hpd$v_found] = false;      ! Initialize help found flag
: 706      1430 2      libraryname [dsc$a_pointer] = librarystring;      ! Initialize descriptor
: 707      1431 2
: 708      1432 2      IF (NOT .subpromptlen)      ! If at TOPIC level
: 709      1433 2          AND ((.help_flags AND hlp$m_all) NEQ 0)      ! and default searching enabled
: 710      1434 2          AND (.keydescs [dsc$w_length] NEQ 0)      ! and keys are non-empty
: 711      1435 2          THEN print_flags [hpd$v_all] = false      ! then print help only if found
: 712      1436 2          ELSE print_flags [hpd$v_all] = true;      ! else print help always
: 713      1437 2
: 714      1438 2      IF NOT (status = call_lbrhelp (last_libindex,
: 715      1439 2          .outputwidth, .printdata, .keydescs))
: 716      1440 2          THEN RETURN (.status);
: 717      1441 2
: 718      1442 2      IF .print_flags [hpd$v_found]      ! If help was found
: 719      1443 2          THEN BEGIN      ! then done
: 720      1444 2          print_flags [hpd$v_found] = false;      ! Reset flag
: 721      1445 2          RETURN true;      ! Exit
: 722      1446 2          END;
: 723      1447 2
: 724      1448 2      print_flags [hpd$v_all] = false;      ! Print help if found
: 725      1449 2
: 726      1450 2      IF .help_flags [hlp$v_library]      ! Check /LIB library
: 727      1451 2          AND (.last_libnumber GEQ 0)      ! if it exists and
: 728      1452 2          THEN BEGIN      ! if haven't already
: 729      1453 2          current_libindex = .main_libindex;      ! Init libindex
: 730      1454 2          IF NOT (status = call_lbrhelp (last_libindex,
: 731      1455 2              .outputwidth, .printdata, .keydescs))
: 732      1456 2              THEN RETURN (.status);
: 733      1457 2          END;
: 734      1458 2
: 735      1459 2      IF .print_flags [hpd$v_found]      ! If help found
: 736      1460 2          THEN current_libnumber = main_libnumber      ! Then set libnumber to main library
: 737      1461 2          ELSE current_libnumber = -1;      ! Else prepare to search default libraries
: 738      1462 2
: 739      1463 2      user_libno = -1;      ! Init library translation
: 740      1464 2      WHILE (NOT .print_flags [hpd$v_found]) AND      ! Search libraries
: 741      1465 2          (tran_next_lib (libraryname, deflib_acmode, user_libno) NEQ 0)
: 742      1466 2      DO BEGIN
: 743      1467 2          current_libnumber = .current_libnumber + 1;      ! Increment current libnumber
: 744      1468 2          IF .current_libnumber NEQ .last_libnumber      ! If lib not already open and searched
: 745      1469 2              THEN IF (open_library (current_libindex, libraryname, .nomsg))      ! If library successfully opened
: 746      1470 2                  THEN BEGIN
: 747      1471 2                      status = call_lbrhelp (current_libindex,
: 748      1472 2                          .outputwidth, .printdata, .keydescs);
: 749      1473 2                      IF NOT .print_flags [hpd$v_found]      ! If help not found
: 750      1474 2                          THEN close_library (current_libindex);      ! Then close library
: 751      1475 2                      IF NOT .status THEN RETURN (.status);      ! If error then return
```

```

: 752      1476 3      END;
: 753      1477 2      END;
: 754      1478 2
: 755      1479 2 IF NOT .print_flags [hpd$w_found]
: 756      1480 3 THEN BEGIN
: 757      1481 3     current_libindex = .last_libindex;
: 758      1482 3     current_libnumber = .last_libnumber;
: 759      1483 3     print_flags [hpd$w_all] = true;
: 760      1484 4     IF NOT (status = call_lbrhelp (last_libindex,
: 761      1485 4         .outputwidth, .printdata, .keydescs))
: 762      1486 3         THEN RETURN (.status);
: 763      1487 2     END;
: 764      1488 2
: 765      1489 3 IF (.current_libnumber NEQ .last_libnumber)
: 766      1490 3 THEN BEGIN
: 767      1491 4     IF (.last_libnumber GTR external_libnumber)
: 768      1492 3         THEN close_library (last_libindex);
: 769      1493 4     IF (.current_libnumber GTR external_libnumber)
: 770      1494 3         THEN
: 771      1495 4         BEGIN
: 772      1496 4             switch_libname (libraryname, .libname);
: 773      1497 4             prompt_flags[hcf$w_more] = true;
: 774      1498 4         END
: 775      1499 3     ELSE
: 776      1500 3         libname [dsc$w_length] = 0;
: 777      1501 3         last_libindex = .current_libindex;
: 778      1502 3         last_libnumber = .current_libnumber;
: 779      1503 2     END;
: 780      1504 2
: 781      1505 2 RETURN true;
: 782      1506 1 END;

```

! Of search_libs

03FC 00000 SEARCH_LIBS:

		59	0000V	CF	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9	1363
		5E	FEEC	CE	9E	00007	MOVAB	CALL_LBRHELP, R9	
57	10	AC		04	C1	0000C	MOVAB	-276(SP), SP	1414
56	10	AC		08	C1	00011	ADDL3	#4, INDICES, R7	1415
		53	0C	AC	D0	00016	ADDL3	#8, INDICES, R6	1416
		52	12	A3	9E	0001A	MOVL	PRINTDATA, R3	
		58		01	D0	0001E	MOVAB	18(R3), R2	1428
		62		04	8A	00021	MOVL	#1, NOMSG	1429
	10	AE	14	AE	9E	00024	BICB2	#4, (R2)	1430
		10	18	AE	9E	00024	MOVAB	LIBRARYSTRING, LIBRARYNAME+4	1432
0A	0000'	CF		0B	E1	0002D	BLBS	SUBPROMPTLEN, 1\$	1433
			04	0B	B5	00033	BBC	#11, HELP_FLAGS, 1\$	1434
				05	13	00036	TSTW	@KEYDESCS	
		62		02	8A	00038	BEQL	1\$	1435
		62		03	11	0003B	BICB2	#2, (R2)	
				02	88	0003D	BRB	2\$	1436
			04	AC	DD	00040	BISB2	#2, (R2)	1439
				53	DD	00043	PUSHL	KEYDESCS	
			08	AC	DD	00045	PUSHL	R3	
							PUSHL	OUTPUTWIDTH	

			57	DD	00048	PUSHL	R7	1438
		69	04	FB	0004A	CALLS	#4, CALL LBRHELP	
		54	50	D0	0004D	MOVL	R0, STATUS	
		2C	54	E9	00050	BLBC	STATUS, 4\$	
06		62	02	E1	00053	BBC	#2, (R2), 3\$	1442
		62	04	8A	00057	BICB2	#4, (R2)	1444
			00D7	31	0005A	BRW	16\$	1445
		62	02	8A	0005D	BICB2	#2, (R2)	1448
1C	0000'	CF	02	E1	00060	BBC	#2, HELP_FLAGS+1, 5\$	1450
			66	D5	00066	TSTL	(R6)	1451
			18	19	00068	BLSS	5\$	
	08	AE	10	BC	D0	MOVL	@INDICES, CURRENT_LIBINDEX	1453
			04	AC	DD	PUSHL	KEYDESCS	1455
				53	DD	PUSHL	R3	
			08	AC	DD	PUSHL	OUTPUTWIDTH	
				57	DD	PUSHL	R7	1454
		69	04	FB	00079	CALLS	#4, CALL LBRHELP	
		54	50	D0	0007C	MOVL	R0, STATUS	
		7E	54	E9	0007F	BLBC	STATUS, 11\$	
05		62	02	E1	00082	BBC	#2, (R2), 6\$	1459
		55	02	CE	00086	MNEGL	#2, CURRENT_LIBNUMBER	1460
			03	11	00089	BRB	7\$	
		55	01	CE	0008B	MNEGL	#1, CURRENT_LIBNUMBER	1461
		6E	01	CE	0008E	MNEGL	#1, USER_LIBNO	1463
6F		62	02	E0	00091	BBS	#2, (R2), 12\$	1464
			5E	DD	00095	PUSHL	SP	1465
			08	AE	9F	PUSHAB	DEFLIB_ACMODE	
			14	AE	9F	PUSHAB	LIBRARYNAME	
	0000V	CF	03	FB	0009D	CALLS	#3, TRAN_NEXT_LIB	
			50	D5	000A2	TSTL	R0	
			39	13	000A4	BEQL	10\$	
			55	D6	000A6	INCL	CURRENT_LIBNUMBER	1467
		66	55	D1	000A8	CMPL	CURRENT_LIBNUMBER, (R6)	1468
			E4	13	000AB	BEQL	8\$	
			58	DD	000AD	PUSHL	NOMSG	1469
			10	AE	9F	PUSHAB	LIBRARYNAME	
			10	AE	9F	PUSHAB	CURRENT_LIBINDEX	
	0000V	CF	03	FB	000B5	CALLS	#3, OPEN_LIBRARY	
		D4	50	E9	000BA	BLBC	R0, 8\$	
			04	AC	DD	PUSHL	KEYDESCS	1472
				53	DD	PUSHL	R3	
			08	AC	DD	PUSHL	OUTPUTWIDTH	
			14	AE	9F	PUSHAB	CURRENT_LIBINDEX	1471
		69	04	FB	000C8	CALLS	#4, CALL LBRHELP	
		54	50	D0	000CB	MOVL	R0, STATUS	
08		62	02	E0	000CE	BBS	#2, (R2), 9\$	1473
			08	AE	9F	PUSHAB	CURRENT_LIBINDEX	1474
	0000V	CF	01	FB	000D5	CALLS	#1, CLOSE_LIBRARY	
		B4	54	E8	000DA	BLBS	STATUS, 8\$	1475
			21	11	000DD	BRB	11\$	
21		62	02	E0	000DF	BBS	#2, (R2), 12\$	1479
			67	D0	000E3	MOVL	(R7), CURRENT_LIBINDEX	1481
	08	AE	66	D0	000E7	MOVL	(R6), CURRENT_LIBNUMBER	1482
		55	02	88	000EA	BISB2	#2, (R2)	1483
		62	04	AC	DD	PUSHL	KEYDESCS	1485
				53	DD	PUSHL	R3	
			08	AC	DD	PUSHL	OUTPUTWIDTH	

		57	DD	000F5	PUSHL	R7		1484
69		04	FB	000F7	CALLS	#4, CALL_LBRHELP		
54		50	D0	000FA	MOVL	R0, STATUS		
04		54	E8	000FD	BLBS	STATUS, 12\$		
50		54	D0	00100	MOVL	STATUS, R0		1486
				04	RET			
66		55	D1	00104	CMPL	CURRENT_LIBNUMBER, (R6)		1489
		2B	13	00107	BEQL	16\$		
		66	D5	00109	TSTL	(R6)		1491
		07	19	0010B	BLSS	13\$		
		57	DD	0010D	PUSHL	R7		1492
0000V	CF	01	FB	0010F	CALLS	#1, CLOSE_LIBRARY		
		55	D5	00114	TSTL	CURRENT_LIBNUMBER		1493
		12	19	00116	BLSS	14\$		
		14	AC	DD	PUSHL	LIBNAME		1496
		10	AE	9F	PUSHAB	LIBRARYNAME		
0000V	CF	02	FB	0011E	CALLS	#2, SWITCH_LIBNAME		
0000'	CF	04	88	00123	BISB2	#4, PROMPT_FLAGS		1497
		03	11	00128	BRB	15\$		1493
		14	BC	B4	CLRW	@LIBNAME		1500
67		08	AE	D0	MOVL	CURRENT_LIBINDEX, (R7)		1501
66		55	D0	00131	MOVL	CURRENT_LIBNUMBER, (R6)		1502
50		01	D0	00134	MOVL	#1, R0		1505
		04	00137	RET				1506

; Routine Size: 312 bytes, Routine Base: \$CODE\$ + 052E


```
: 784      1507 1 %SBTTL 'Routine change lib';
: 785      1508 1 ROUTINE change_lib (getcddesc, charpos, indices, libname) =
: 786      1509 2 BEGIN
: 787      1510 2
: 788      1511 2 ++
: 789      1512 2 FUNCTIONAL DESCRIPTION:
: 790      1513 2
: 791      1514 2     This routine changes the library context currently in effect to
: 792      1515 2     the library specified by the command descriptor. It also removes
: 793      1516 2     the library specification from the command string.
: 794      1517 2
: 795      1518 2 INPUTS:
: 796      1519 2
: 797      1520 2    getcddesc = address of descriptor containing the new library name
: 798      1521 2
: 799      1522 2     charpos = pointer to '@' preceeding library name
: 800      1523 2
: 801      1524 2     indices = address of data structure containing indices of
: 802      1525 2     libraries currently in use
: 803      1526 2
: 804      1527 2     libname = address of descriptor for the user default library name
: 805      1528 2
: 806      1529 2 OUTPUTS:
: 807      1530 2
: 808      1531 2    getcddesc = library specification is removed from string
: 809      1532 2
: 810      1533 2     indices = if new library successfully found, updated to
: 811      1534 2     reflect library that was specified by the
: 812      1535 2     command string
: 813      1536 2
: 814      1537 2     libprompt = if new library successfully found, updated to
: 815      1538 2     the file name of that library
: 816      1539 2
: 817      1540 2 ROUTINE VALUE:
: 818      1541 2
: 819      1542 2     True, if new library found.
: 820      1543 2     False, if library unchanged.
: 821      1544 2 --
: 822      1545 2 MAP
: 823      1546 2     indices : REF BBLOCK,
: 824      1547 2    getcddesc : REF BBLOCK;
: 825      1548 2
: 826      1549 2 BIND
: 827      1550 2     last_libindex = indices [hli$l_lastindex],
: 828      1551 2     last_libnumber = indices [hli$_lastnumb];
: 829      1552 2
: 830      1553 2 LOCAL
: 831      1554 2     current_libindex,
: 832      1555 2     current_libnumber,
: 833      1556 2     deflib_acmode : BYTE,
: 834      1557 2     librarystring : BBLOCK [nam$c_maxrss],
: 835      1558 2     libraryname : BBLOCK [dsc$c_s_bln],
: 836      1559 2     name_end,
: 837      1560 2     name_len,
: 838      1561 2     nomsg,
: 839      1562 2     temp_end,
: 840      1563 2     user_libno;

! Index of last library examined
! No. of last library examined, relative to all defa
! Index of library currently being searched
! Number of library currently being searched
! Logical name table number
! Default library name string
! String descriptor for library name
! End of @ string
! Length of @ string
! Open library message flag
! Location of first '/' in command string
! HLP$LIBRARY number
```

```

: 841      1564      2
: 842      1565      2 remove_terminator (.getcddesc);
: 843      1566
: 844      1567      2 IF (.charpos + 1) EQL
: 845      1568      3 (.getcddesc [dsc$w_length] + .getcddesc [dsc$a_pointer])
: 846      1569      3 THEN RETURN false;
: 847      1570
: 848      1571      2 nomsg = true;
: 849      1572      2 libraryname [dsc$a_pointer] = librarystring;
: 850      1573      2 make_upper_case (.getcddesc, .getcddesc [dsc$a_pointer]);
: 851      1574
: 852      1575      2 name_len = .getcddesc [dsc$w_length] +
: 853      1576      3 .getcddesc [dsc$a_pointer] - .charpos;
: 854      1577      2 IF (name_end = CH$FIND CH (.name_len, .charpos, '%C' ')) EQL 0
: 855      1578      3 THEN name_end = .getcddesc [dsc$w_length]
: 856      1579      4 + .getcddesc [dsc$a_pointer];
: 857      1580      2 IF (temp_end = CH$FIND CH (.name_len, .charpos, '%C' '/')) NEQ 0
: 858      1581      3 THEN IF .temp_end LSS .name_end
: 859      1582      4 THEN name_end = .temp_end;
: 860      1583      2 name_len = .name_end - .charpos;
: 861      1584
: 862      1585      2 current_libnumber = -1;
: 863      1586      2 user_libno = -1;
: 864      1587      2 WHILE (tran_next_lib (libraryname, deflib_acmode, user_libno)
: 865      1588      3 EQL true)
: 866      1589      2 DO BEGIN
: 867      1590      3 LOCAL
: 868      1591      4 filename : BBLOCK [dsc$c_s_bln],
: 869      1592      4 filebuffer : VECTOR [filen_name_length, BYTE];
: 870      1593
: 871      1594      4 filename = 0;
: 872      1595      4 filename [dsc$a_pointer] = filebuffer;
: 873      1596      4 switch_libname (libraryname, filename);
: 874      1597
: 875      1598      4 current_libnumber = .current_libnumber + 1;
: 876      1599
: 877      1600      4 IF CH$EQL (.name_len - 1, .filename [dsc$a_pointer],
: 878      1601      5 .name_len - 1, .charpos + 1, 0)
: 879      1602      5
: 880      1603      4 THEN IF (open_library (current_libindex, libraryname, .nomsg))
: 881      1604      5 THEN BEGIN
: 882      1605      6 IF .last_libnumber GEQ external_libnumber
: 883      1606      7 THEN close_library (last_libindex);
: 884      1607      6 switch_libname (libraryname, .libname);
: 885      1608      6 last_libindex = .current_libindex;
: 886      1609      6 last_libnumber = .current_libnumber;
: 887      1610      6 getcddesc [dsc$w_length] =
: 888      1611      7 .getcddesc [dsc$w_length] - .name_len;
: 889      1612      6 getcddesc [dsc$a_pointer] =
: 890      1613      7 .getcddesc [dsc$a_pointer] + .name_len;
: 891      1614      6 RETURN true;
: 892      1615      5 END;
: 893      1616
: 894      1617      2 END;
: 895      1618
: 896      1619      2 !
: 897      1620      2 ! If library file name was not found in logical name tables,
```

```

! Remove command terminator
! If no file name
! Then return false
! Do not signal error if library can
! Initialize local descriptor
! Upper case command
! Calculate length of @ to end of li
! If no blank in command string
! Then end is end of command
! If '/' position is before blank po
! Then end is position of '/'
! Calculate length of @ string
! Init library number
! Init default library searching
! Incr libnumber
! Is command a substring of the file
! If library successfully opened
! If last library not main library
! Then close it
! Change library name
! Set libindex
! Set libnumber
! Remove @ string from command
! Return success
! Library not found
```



```

: 898      1621 2 ! then assume that the file name is actually a full file spec
: 899      1622 2 ! for a library that exists but is not a user-defined default
: 900      1623 2 ! library. Try to open that library.
: 901      1624 2 !
: 902      1625 2
: 903      1626 2 libraryname [dsc$w_length] = .name_len - 1;           ! Initialize library name
: 904      1627 2 libraryname [dsc$a_pointer] = .charpos + 1;
: 905      1628 2
: 906      1629 3 IF NOT (open library (current_libindex, libraryname, .nomsg)) ! Try opening library
: 907      1630 2 THEN RETURN false;                                     ! If unsuccessful then give up and s
: 908      1631 2
: 909      1632 2 IF .last_libnumber GEQ external_libnumber             ! If last library not main library
: 910      1633 2 THEN close_library (last_libindex);                 ! Then close it
: 911      1634 2 switch_libname (libraryname, .libname);              ! Change library name
: 912      1635 2 last_libindex = .current_libindex;                   ! Set libindex
: 913      1636 2 last_libnumber = -1;                                   ! Set libnumber
: 914      1637 2 getcmddesc [dsc$w_length] = .getcmddesc [dsc$w_length] - .name_len; ! Remove @ string from command
: 915      1638 2 getcmddesc [dsc$a_pointer] = .getcmddesc [dsc$a_pointer] + .name_len;
: 916      1639 2
: 917      1640 2 RETURN true;                                           ! Return success
: 918      1641 1 END;                                                  ! Of change_lib
```

OFFC 00000 CHANGE_LIB:

		5E	FEB	CE	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1508
		AC		04	C1	00007	MOVAB	-324(SP), SP	
5B	OC	AC		08	C1	0000C	ADDL3	#4, INDICES, R11	1550
5A	OC	AC					ADDL3	#8, INDICES, R10	1551
		55	04	AC	D0	00011	MOVL	GETCMDDESC, R5	1565
				55	DD	00015	PUSHL	R5	
	0000V	CF		01	FB	00017	CALLS	#1, REMOVE TERMINATOR	
		56	08	AC	D0	0001C	MOVL	CHARPOS, R6	1567
		51	01	A6	9E	00020	MOVAB	1(R6), R1	
		50		65	3C	00024	MOVZWL	(R5), R0	1568
		50	04	A5	C0	00027	ADDL2	4(R5), R0	
		50		51	D1	0002B	CMPL	R1, R0	
				03	12	0002E	BNEQ	1\$	
				00F8	31	00030	BRW	11\$	
		59		01	D0	00033	MOVL	#1, NOMSG	1571
	40	AE	44	AE	9E	00036	MOVAB	LIBRARYSTRING, LIBRARYNAME+4	1572
			04	A5	DD	0003B	PUSHL	4(R5)	1573
				55	DD	0003E	PUSHL	R5	
	0000V	CF		02	FB	00040	CALLS	#2, MAKE_UPPER_CASE	
		53		65	3C	00045	MOVZWL	(R5), R3	1576
		53	04	A5	C0	00048	ADDL2	4(R5), R3	
54		53		56	C3	0004C	SUBL3	R6, R3, NAME_LEN	
66		54		20	3A	00050	LOCC	#32, NAME_LEN, (R6)	1577
				02	12	00054	BNEQ	2\$	
				51	D4	00056	CLRL	R1	
		52		51	D0	00058	MOVL	R1, NAME_END	
				03	12	0005B	BNEQ	3\$	
		52		53	D0	0005D	MOVL	R3, NAME_END	1579
66		54		2F	3A	00060	LOCC	#47, NAME_LEN, (R6)	1580
				02	12	00064	BNEQ	4\$	

			51	D4	00066		CLRL	R1		
			51	D5	00068	4\$:	TSTL	TEMP_END		
			08	13	0006A		BEQL	5\$		
		52	51	D1	0006C		CMPL	TEMP_END, NAME_END	1581	
			03	18	0006F		BGEQ	5\$		
		52	51	D0	00071		MOVL	TEMP_END, NAME_END	1582	
54		52	56	C3	00074	5\$:	SUBL3	R6, NAME_END, NAME_LEN	1583	
		58	01	CE	00078		MNEGL	#1, CURRENT_LIBNUMBER	1585	
		6E	01	CE	0007B		MNEGL	#1, USER_LIBNO	1586	
		57	FF	A4	9E	0007E	MOVAB	-1(R4), R7	1600	
				5E	DD	00082	PUSHL	SP	1587	
			08	AE	9F	00084	PUSHAB	DEFLIB_ACMODE		
			44	AE	9F	00087	PUSHAB	LIBRARYNAME		
	0000V	CF	03	FB	0008A		CALLS	#3, TRAN_NEXT_LIB		
		01	50	D1	0008F		CMPL	R0, #1	1588	
			51	12	00092		BNEQ	8\$		
			34	AE	D4	00094	CLRL	FILENAME	1594	
	38	AE	0C	AE	9E	00097	MOVAB	FILEBUFFER, FILENAME+4	1595	
			34	AE	9F	0009C	PUSHAB	FILENAME	1596	
			40	AE	9F	0009F	PUSHAB	LIBRARYNAME		
	0000V	CF	02	FB	000A2		CALLS	#2, SWITCH_LIBNAME		
			58	D6	000A7		INCL	CURRENT_LIBNUMBER	1598	
01	A6	38	57	29	000A9		CMPC3	R7, @FILENAME+4, 1(R6)	1600	
			D1	12	000AF		BNEQ	6\$		
			59	DD	000B1		PUSHL	NOMSG	1603	
			40	AE	9F	000B3	PUSHAB	LIBRARYNAME		
			10	AE	9F	000B6	PUSHAB	CURRENT_LIBINDEX		
	0000V	CF	03	FB	000B9		CALLS	#3, OPEN_LIBRARY		
		C1	50	E9	000BE		BLBC	R0, 6\$		
	FFFFFFFF	8F	6A	D1	000C1		CMPL	(R10), #-1	1605	
			07	19	000C8		BLSS	7\$		
			5B	DD	000CA		PUSHL	R11	1606	
	0000V	CF	01	FB	000CC		CALLS	#1, CLOSE_LIBRARY		
			10	AC	DD	000D1	PUSHL	LIBNAME	1607	
			40	AE	9F	000D4	PUSHAB	LIBRARYNAME		
	0000V	CF	02	FB	000D7		CALLS	#2, SWITCH_LIBNAME		
		6B	58	D0	000DC		MOVL	CURRENT_LIBINDEX, (R11)	1608	
		6A	58	D0	000E0		MOVL	CURRENT_LIBNUMBER, (R10)	1609	
			3B	11	000E3		BRB	10\$	1611	
	3C	AE	57	B0	000E5	8\$:	MOVW	R7, LIBRARYNAME	1626	
	40	AE	A6	9E	000E9		MOVAB	1(R6), LIBRARYNAME+4	1627	
			59	DD	000EE		PUSHL	NOMSG	1629	
			40	AE	9F	000F0	PUSHAB	LIBRARYNAME		
			10	AE	9F	000F3	PUSHAB	CURRENT_LIBINDEX		
	0000V	CF	03	FB	000F6		CALLS	#3, OPEN_LIBRARY		
		2D	50	E9	000FB		BLBC	R0, 11\$		
	FFFFFFFF	8F	6A	D1	000FE		CMPL	(R10), #-1	1632	
			07	19	00105		BLSS	9\$		
			5B	DD	00107		PUSHL	R11	1633	
	0000V	CF	01	FB	00109		CALLS	#1, CLOSE_LIBRARY		
			10	AC	DD	0010E	PUSHL	LIBNAME	1634	
			40	AE	9F	00111	PUSHAB	LIBRARYNAME		
	0000V	CF	02	FB	00114		CALLS	#2, SWITCH_LIBNAME		
		6B	58	D0	00119		MOVL	CURRENT_LIBINDEX, (R11)	1635	
		6A	01	CE	0011D		MNEGL	#1, (R10)	1636	
		65	54	A2	00120	10\$:	SUBW2	NAME_LEN, (R5)	1637	
	04	A5	54	C0	00123		ADDL2	NAME_LEN, 4(R5)	1638	

LBR_OUTPUTHELP	Prompting and library searching help function	M 14	16-Sep-1984 02:04:00	VAX-11 Bliss-32 V4.0-742	Page 29
V04=000	Routine change_lib		14-Sep-1984 12:37:45	[LBR.SRC]OUTPUTHELP.B32;1	(6)
	50	01 D0 00127	MOVL #1, R0		: 1640
		04 0012A	RET		: 1641
	50	04 0012B 11\$:	CLRL R0		: 1641
		04 0012D	RET		: 1641

; Routine Size: 302 bytes, Routine Base: \$CODE\$ + 0666

```
: 920      1642 1 %SBTTL 'Routine switch_libname';
: 921      1643 1 ROUTINE switch_libname(newname, oldname) =
: 922      1644 2 BEGIN
: 923      1645 2
: 924      1646 2 !++
: 925      1647 2 FUNCTIONAL DESCRIPTION:
: 926      1648 2
: 927      1649 2     This routine inserts a new libname into the old descriptor.
: 928      1650 2
: 929      1651 2 INPUTS:
: 930      1652 2
: 931      1653 2     newname =      address of a descriptor for the new library file name
: 932      1654 2
: 933      1655 2     oldname =     address of a descriptor for the old library file name
: 934      1656 2
: 935      1657 2 OUTPUTS:
: 936      1658 2
: 937      1659 2     oldname =     updated to specify the name of the new library
: 938      1660 2
: 939      1661 2 ROUTINE VALUE:
: 940      1662 2
: 941      1663 2     Always true.
: 942      1664 2
: 943      1665 2 --
: 944      1666 2 MAP
: 945      1667 2     newname : REF BBLOCK,
: 946      1668 2     oldname : REF BBLOCK;
: 947      1669 2
: 948      1670 2 LOCAL
: 949      1671 2     rsabuf: BBLOCK [nam$c_maxrss],      ! buffer for resultant string
: 950      1672 2     esabuf: BBLOCK [nam$c_maxrss],      ! buffer for expanded string
: 951      1673 2     libfab: BBLOCK [fab$c_bln],          ! temporary FAB
: 952      1674 2     libnam: BBLOCK [nam$c_bln];        ! temporary NAM block
: 953      1675 2
: 954      1676 2 $NAM_INIT( NAM=libnam,
: 955      1677 2     RSA=rsabuf,
: 956      1678 2     RSS=nam$c_maxrss,
: 957      1679 2     ESA=esabuf,
: 958      1680 2     ESS=nam$c_maxrss);
: 959      1681 2
: 960      1682 2 $FAB_INIT( FAB=libfab,
: 961      1683 2     FOP=NAM,
: 962      1684 2     FNA=.newname[dsc$a_pointer],
: 963      1685 2     FNS=.newname[dsc$w_length],
: 964      1686 2     NAM=libnam);
: 965      1687 2
: 966      1688 2 $PARSE(FAB=libfab);
: 967      1689 2
: 968      1690 2     oldname[dsc$w_length] = .libnam[nam$b_name];
: 969      1691 2     CH$MOVE(.libnam[nam$b_name], .libnam[nam$l_name], .oldname[dsc$a_pointer]);
: 970      1692 2
: 971      1693 2 RETURN true;
: 972      1694 1 END;
```

! Of switch_libname

.EXTRN SYSSPARSE

				003C 00000 SWITCH_LIBNAME:						
0060	8F	00	5E	FD50	CE	9E	00002	WORD	Save R2,R3,R4,R5	: 1643
			6E		00	2C	00007	MOVAB	-688(SP), SP	: 1680
					6E		0000E	MOVCS	#0, (SP), #0, #96, \$RMS_PTR	
		02	6E	6002	8F	B0	0000F	MOVW	#24578, \$RMS_PTR	
		04	AE		01	8E	00014	MNEGB	#1, \$RMS_PTR+2	
		0A	AE	FF00	CD	9E	00018	MOVAB	RSABUF, \$RMS_PTR+4	
		0C	AE		01	8E	0001E	MNEGB	#1, \$RMS_PTR+10	
0050	8F	00	AE	00B0	CE	9E	00022	MOVAB	ESABUF, \$RMS_PTR+12	
			6E		00	2C	00028	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	: 1686
				60	AE		0002F			
		60	AE	5003	8F	B0	00031	MOVW	#20483, \$RMS_PTR	
		64	AE	01000000	8F	D0	00037	MOVL	#16777216, \$RMS_PTR+4	
		76	AE		02	90	0003F	MOVB	#2, \$RMS_PTR+22	
		7F	AE		02	90	00043	MOVB	#2, \$RMS_PTR+31	
		0088	CE		6E	9E	00047	MOVAB	LIBNAM, \$RMS_PTR+40	
			50	04	AC	D0	0004C	MOVL	NEWNAME, R0	
		008C	CE	04	A0	D0	00050	MOVL	4(R0), \$RMS_PTR+44	
		0094	CE		60	90	00056	MOVB	(R0), \$RMS_PTR+52	
				60	AE	9F	0005B	PUSHAB	LIBFAB	: 1688
		00000000G	00		01	FB	0005E	CALLS	#1, SYSSPARSE	
			50	08	AC	D0	00065	MOVL	OLDNAME, R0	: 1690
			60	3B	AE	9B	00069	MOVZBW	LIBNAM+59, (R0)	
			51	3B	AE	9A	0006D	MOVZBL	LIBNAM+59, R1	: 1691
	04	B0	4C		51	28	00071	MOVCS	R1, @LIBNAM+76, @4(R0)	
			50		01	D0	00077	MOVL	#1, R0	: 1693
					04	00	0007A	RET		: 1694

; Routine Size: 123 bytes, Routine Base: \$CODE\$ + 0794


```

: 974      1695 1 %SBTTL 'Routine tran_next_lib';
: 975      1696 1 ROUTINE tran_next_lib (libname, deflibacmode, userlibno) =
: 976      1697 2 BEGIN
: 977      1698 2
: 978      1699 2 ++
: 979      1700 2 FUNCTIONAL DESCRIPTION:
: 980      1701 2
: 981      1702 2     This routine returns the value true if a default
: 982      1703 2     library if found and false if not. If a library is found,
: 983      1704 2     the library file name is returned in the descriptor libname.
: 984      1705 2
: 985      1706 2 INPUTS:
: 986      1707 2
: 987      1708 2     libname =      address of a descriptor to return the library name in
: 988      1709 2
: 989      1710 2     deflibacmode = address of a longword containing the access mode of the
: 990      1711 2     last logical name table searched
: 991      1712 2
: 992      1713 2     userlibno =    address of a longword containing the number of the last
: 993      1714 2     library found in the last logical name table. If the
: 994      1715 2     number is -1, then start searching the logical name
: 995      1716 2     tables all over again.
: 996      1717 2
: 997      1718 2 OUTPUTS:
: 998      1719 2
: 999      1720 2     libname, deflibacmode, userlibno : as described above
1000      1721 2
1001      1722 2 ROUTINE VALUE:
1002      1723 2
1003      1724 2     True, if user default library is found.
1004      1725 2     False, if no more user default libraries left, i.e., at end of tables.
1005      1726 2
1006      1727 2 --
1007      1728 2
1008      1729 2 MAP
1009      1730 2     help_flags : BITVECTOR,
1010      1731 2     libname : REF BBLOCK;
1011      1732 2
1012      1733 2 BIND
1013      1734 2     maxliblen = %CHARCOUNT('HLP$LIBRARY_999');           ! Max lib logical name length
1014      1735 2
1015      1736 2 OWN
1016      1737 2     hlplibnam : COUNTEDSTRING ('HLP$LIBRARY'),                ! Initial library logical name
1017      1738 2     libnamefao : COUNTEDSTRING ('HLP$LIBRARY_!UW');           ! FAO descriptor for general library logical name
1018      1739 2
1019      1740 2 LOCAL
1020      1741 2     deflibdsbmsk : VECTOR [4,BYTE],                             ! Logical name table access modes
1021      1742 2     deflibnam : VECTOR [maxliblen,BYTE],                         ! Library logical name buffer
1022      1743 2     deflibdesc : BBLOCK [dsc$c_s_bln],                          ! Library logical name descriptor
1023      1744 2     status;                                                     ! Status of logical name translation
1024      1745 2
1025      1746 2 IF ..userlibno LSS 0                                           ! If starting search at beginning
1026      1747 2 THEN BEGIN                                                    ! Then initialize access mode and li
1027      1748 2     .deflibacmode = 0;
1028      1749 2     .userlibno = 0;
1029      1750 2     END;
1030      1751 2
```



```
: 1031      1752  2 deflibdesc [dsc$w_length] = maxliblen;           ! Initialize logical name descriptor
: 1032      1753  2 deflibdesc [dsc$a_pointer] = deflibnam;
: 1033      1754  2
: 1034      1755  2 IF ..userlibno EQL 0                               ! If first logical name in table
: 1035      1756  2 THEN BEGIN                                       ! Then special case, do not use FAO
: 1036      1757  3   deflibdesc [dsc$w_length] = .hlplibnam [0];
: 1037      1758  3   CH$MOVE (.deflibdesc [dsc$w_length], hlplibnam + 1, deflibnam);
: 1038      1759  3   END
: 1039      1760  3
: 1040      1761  3 ELSE BEGIN                                       ! Else use FAO to combine logical na
: 1041      1762  3   deflibdesc [dsc$w_length] = .libnamefao [0];
: 1042      1763  3   CH$MOVE (.deflibdesc [dsc$w_length], libnamefao + 1, deflibnam);
: 1043      1764  3   IF NOT $FAO (deflibdesc, deflibdesc, deflibdesc, ..userlibno)
: 1044      1765  3   THEN RETURN false;
: 1045      1766  3   END;
: 1046      1767  2
: 1047      1768  2 IF .help_flags [..deflibacmode + 1] NEQ 0       ! If searching of current logical na
: 1048      1769  2 THEN BEGIN                                       ! Then translate logical name for th
: 1049      1770  3   deflibdsbmsk = %X'00060503';
: 1050      1771  3   libname [dsc$w_length] = nam$c_maxrss;
: 1051      1772  3   status = $TRNLOG (LOGNAM = deflibdesc,
P      1773  3   DSBMSK = .deflibdsbmsk [..deflibacmode],
P      1774  3   RSLLEN = .libname,
P      1775  3   RSLBUF = .libname);
: 1055      1776  4   IF .status AND (.status NEQ SS$_NOTRAN)       ! If logical name is successfully tr
: 1056      1777  4   THEN BEGIN                                       ! Then increment lib no. for next se
: 1057      1778  4   .userlibno = ..userlibno + 1;
: 1058      1779  4   RETURN true;
: 1059      1780  4   END;
: 1060      1781  3   END;
: 1061      1782  2
: 1062      1783  2 ! If current logical name table not enabled or logical name unsuccessfully
: 1063      1784  2 ! translated, then recursively call this routine to search next table.
: 1064      1785  2
: 1065      1786  2
: 1066      1787  2
: 1067      1788  2 IF (.deflibacmode = ..deflibacmode + 1) GTR 2   ! Increment access mode
: 1068      1789  2 THEN RETURN false;                                ! If out of tables, then return fals
: 1069      1790  2 .userlibno = 0;                                    ! Reset lib no. to start of table
: 1070      1791  2 RETURN tran_next_lib (.libname, .deflibacmode, .userlibno); ! Make recursive call
: 1071      1792  2
: 1072      1793  1 END;                                           ! Of tran_next_lib
```

.PSECT \$OWNS,NOEXE,2

OB 00038 HLPLIBNAM:

.BYTE 11

.ASCII \HLP\$LIBRARY\

OF 00044 LIBNAMEFAO:

.BYTE 15

.ASCII \HLP\$LIBRARY_!UW\

MAXLIBLEN=

15

.EXTRN SYSS\$FAO, SYS\$TRNLOG

57 55 21 5F 59 52 41 52 42 49 4C 24 50 4C 48 00045

				.PSECT		SCODE\$,NOWRT,2			
				00FC 00000		TRAN_NEXT LIB:			
		57	0000'	CF	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7	1696
		5E		1C	C2	00007	MOVAB	HLPLIBNAM, R7	
		56	0C	AC	D0	0000A	SUBL2	#28, SP	1746
				66	D5	0000E	MOVL	USERLIBNO, R6	
				05	18	00010	TSTL	(R6)	
			08	BC	D4	00012	BGEQ	1\$	1748
				66	D4	00015	CLRL	@DEFLIBACMODE	1749
	04	AE		0F	B0	00017	CLRL	(R6)	1752
	08	AE	0C	AE	9E	0001B	MOVW	#15, DEFLIBDESC	1753
				66	D5	00020	MOVAB	DEFLIBNAM, DEFLIBDESC+4	1755
				0D	12	00022	TSTL	(R6)	
		04	AE	67	9B	00024	BNEQ	2\$	
OC	AE	01	A7	04	AE	28	MOVZBW	HLPLIBNAM, DEFLIBDESC	1757
					21	11	MOVW	DEFLIBDESC, HLPLIBNAM+1, DEFLIBNAM	1758
		04	AE	0C	A7	9B	BRB	3\$	1755
OC	AE	0D	A7	04	AE	28	MOVZBW	LIBNAMEFAO, DEFLIBDESC	1762
					66	DD	MOVW	DEFLIBDESC, LIBNAMEFAO+1, DEFLIBNAM	1763
					08	AE	PUSHL	(R6)	1764
					0C	AE	PUSHAB	DEFLIBDESC	
					10	AE	PUSHAB	DEFLIBDESC	
		00000000G	00	04	FB	00048	PUSHAB	DEFLIBDESC	
		60		50	E9	0004F	CALLS	#4, SYSSFAO	
		50	08	BC	D0	00052	BLBC	R0, 5\$	
		52	01	A0	9E	00056	MOVL	@DEFLIBACMODE, R0	1768
51	F8	A7	01	52	EF	0005A	MOVAB	1(R0), R2	
				34	13	00060	EXTZV	R2, #1, HELP_FLAGS, R1	
				8F	D0	00062	BEQL	4\$	
		04	6E	00060503	8F	9B	MOVL	#394499, DEFLIBDSBMSK	1770
			BC	FF	9A	00069	MOVZBW	#255, @LIBNAME	1771
			7E	6E40	9A	0006E	MOVZBL	DEFLIBDSBMSK[R0], -(SP)	1775
				7E	7C	00072	CLRQ	-(SP)	
				04	AC	DD	PUSHL	LIBNAME	
				04	AC	DD	PUSHL	LIBNAME	
				18	AE	9F	PUSHAB	DEFLIBDESC	
		00000000G	00	06	FB	0007D	CALLS	#6, SYSTRNLOG	
			0F	50	E9	00084	BLBC	STATUS, 4\$	1776
		00000629	8F	50	D1	00087	CMPL	STATUS, #1577	
				06	13	0008E	BEQL	4\$	
				66	D6	00090	INCL	(R6)	1778
			50	01	D0	00092	MOVL	#1, R0	1779
					04	00095	RET		
	50	08	BC	01	C1	00096	ADDL3	#1, @DEFLIBACMODE, R0	1788
		08	BC	50	D0	0009B	MOVL	R0, @DEFLIBACMODE	
			02	50	D1	0009F	CMPL	R0, #2	
				0E	14	000A2	BGTR	5\$	
				66	D4	000A4	CLRL	(R6)	1790
				56	DD	000A6	PUSHL	R6	1791
		7E	04	AC	7D	000AB	MOVQ	LIBNAME, -(SP)	
		FF4F	CF	03	FB	000AC	CALLS	#3, TRAN_NEXT_LIB	
					04	000B1	RET		
				50	D4	000B2	CLRL	R0	1793
					04	000B4	RET		

; Routine Size: 181 bytes, Routine Base: \$CODE\$ + 080F

LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine tran_next_lib

F 15
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 35
(8)

LB
V0

```
: 1074 1794 1 %SBTTL 'Routine open_library';
: 1075 1795 1 ROUTINE open_library(libindex, libname, nomsgflag) =
: 1076 1796 2 BEGIN
: 1077 1797 2
: 1078 1798 2 |++
: 1079 1799 2 | FUNCTIONAL DESCRIPTION:
: 1080 1800 2 |
: 1081 1801 2 |     Open the library with the specified name and return its index.
: 1082 1802 2 |
: 1083 1803 2 | INPUTS:
: 1084 1804 2 |
: 1085 1805 2 |     libindex =      address of longword to contain index of opened library
: 1086 1806 2 |
: 1087 1807 2 |     libname =       address of descriptor for library name
: 1088 1808 2 |
: 1089 1809 2 |     nomsgflag =     flag that is true if open errors should not be signalled
: 1090 1810 2 |
: 1091 1811 2 | OUTPUTS:
: 1092 1812 2 |
: 1093 1813 2 |     libindex : as described above
: 1094 1814 2 |
: 1095 1815 2 | ROUTINE VALUE:
: 1096 1816 2 |
: 1097 1817 2 |     True, if library successfully opened.
: 1098 1818 2 |     False, if unsuccessful.
: 1099 1819 2 |
: 1100 1820 2 | --
: 1101 1821 2 |
: 1102 1822 2 | MAP
: 1103 1823 2 |     libname : REF BBLOCK;
: 1104 1824 2 |
: 1105 1825 2 | EXTERNAL
: 1106 1826 2 |     lbr$gl_rmsstv : ADDRESSING_MODE (GENERAL);           ! RMS STV from librarian
: 1107 1827 2 |
: 1108 1828 2 | LOCAL
: 1109 1829 2 |     filnamdesc : BBLOCK [dsc$c_s_bln],                   ! Library name descriptor
: 1110 1830 2 |     help_defname : BBLOCK [dsc$c_s_bln],                 ! Default filename descriptor
: 1111 1831 2 |     helpfilename : BBLOCK [nam$c_maxrss],                ! Resultant file name
: 1112 1832 2 |     help_func,                                           ! Library access type
: 1113 1833 2 |     help_type,                                           ! Type of library
: 1114 1834 2 |     namblk : BBLOCK [nam$c_bln],                         ! Library name block
: 1115 1835 2 |     status;
: 1116 1836 2 |
: 1117 1837 2 | $NAM_INIT (NAM = namblk,                                  ! Initialize the NAM block
: 1118 1838 2 |             RSS = nam$c_maxrss,
: 1119 1839 2 |             RSA = helpfilename,
: 1120 1840 2 |             ESS = nam$c_maxrss,
: 1121 1841 2 |             ESA = helpfilename);
: 1122 1842 2 |
: 1123 1843 2 | help_func = lbr$c_read;                                   ! Will be reading the library
: 1124 1844 2 | help_type = lbr$c_typ_hlp;                               ! Library is of help type
: 1125 1845 2 |
: 1126 1846 2 |
: 1127 1847 2 | ! Call librarian to initialize control. Stop if error.
: 1128 1848 2 |
: 1129 1849 2 |
: 1130 1850 3 IF NOT (status = lbr$ini_control (.libindex, help_func, help_type, namblk))
```



```

1131 1851 2 THEN
1132 1852 3 BEGIN
1133 1853 3 SIGNAL (.status);
1134 1854 3 RETURN .status OR sts$m_inhib_msg;
1135 1855 3 END;
1136 1856 3
1137 1857 2 !
1138 1858 2 ! Call librarian to open library. Stop if error should be flagged.
1139 1859 2 !
1140 1860 2
1141 1861 2 help_defname [dsc$w_length] = .syshelp [0]; ! Initialize default library directory and type
1142 1862 2 help_defname [dsc$a_pointer] = syshelp [1];
1143 1863 2
1144 1864 3 IF NOT (status = lbr$open (.libindex, .libname, 0, help_defname))
1145 1865 2 AND NOT .nomsgflag
1146 1866 3 THEN BEGIN
1147 1867 3 IF (filnamdesc [dsc$w_length] = .namblk [nam$b_esl]) NEQ 0
1148 1868 3 THEN filnamdesc [dsc$a_pointer] = .namblk [nam$l_esa]
1149 1869 4 ELSE BEGIN
1150 1870 4 filnamdesc [dsc$w_length] = .libname [dsc$w_length];
1151 1871 4 filnamdesc [dsc$a_pointer] = .libname [dsc$a_pointer];
1152 1872 3 END;
1153 1873 3 SIGNAL (shr$_openin OR hlp$c_facility OR sts$k_error,
1154 1874 3 1, filnamdesc, .status, .lbr$g[rmsstv]);
1155 1875 3 status = shr$_openin OR hlp$c_facility OR sts$k_error OR sts$m_inhib_msg;
1156 1876 2 END;
1157 1877 2
1158 1878 2 RETURN .status;
1159 1879 2
1160 1880 1 END; !Of open_library

```

.EXTRN LBR\$GL_RMSSTV

007C 00000 OPEN_LIBRARY:

[illegible]

LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine open_library

I 15
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 38
(9)

50		53 10000000	8F C9 00054	BISL3 #268435456, STATUS, R0	: 1854
			04 0005C	RET	:
F0	AD	0000'	CF 9B 0005D 1\$:	MOVZBW SYSHELP, HELP_DEFNAME	: 1861
F4	AD	0000'	CF 9E 00063	MOVAB SYSHELP+1, HELP_DEFNAME+4	: 1862
		F0	AD 9F 00069	PUSHAB HELP_DEFNAME	: 1864
			7E D4 0006C	CLRL -(SP)	:
		52 08	AC D0 0006E	MOVL LIBNAME, R2	:
			52 DD 00072	PUSHL R2	:
		04	AC DD 00074	PUSHL LIBINDEX	:
00000000G	00		04 FB 00077	CALLS #4, LBR\$OPEN	:
	53		50 D0 0007E	MOVL R0, STATUS	:
	38		53 E8 00081	BLBS STATUS, 4\$:
	34	0C	AC E8 00084	BLBS NOMSGFLAG, 4\$: 1865
F8	AD	13	AE 9B 00088	MOVZBW NAMBLK+11, FILNAMDESC	: 1867
			07 13 0008D	BEQL 2\$:
FC	AD	14	AE D0 0008F	MOVL NAMBLK+12, FILNAMDESC+4	: 1868
			09 11 00094	BRB 3\$:
F8	AD		62 B0 00096 2\$:	MOVW (R2), FILNAMDESC	: 1870
FC	AD	04	A2 D0 0009A	MOVL 4(R2), FILNAMDESC+4	: 1871
		00000000G	00 DD 0009F 3\$:	PUSHL LBR\$GL_RMSSTV	: 1874
			53 DD 000A5	PUSHL STATUS	:
		F8	AD 9F 000A7	PUSHAB FILNAMDESC	: 1873
			01 DD 000AA	PUSHL #1	:
		0076109A	8F DD 000AC	PUSHL #7737498	:
66			05 FB 000B2	CALLS #5, LIB\$SIGNAL	:
53 1076109A			8F D0 000B5	MOVL #276172954, STATUS	: 1875
50			53 D0 000BC 4\$:	MOVL STATUS, R0	: 1878
			04 000BF	RET	: 1880

; Routine Size: 192 bytes, Routine Base: \$CODE\$ + 08C4

LB
V0


```
: 1162      1881 1 %SBTTL 'Routine close_library';
: 1163      1882 1 ROUTINE close_library(libindex) =
: 1164      1883 2 BEGIN
: 1165      1884 2
: 1166      1885 2 !++
: 1167      1886 2 FUNCTIONAL DESCRIPTION:
: 1168      1887 2
: 1169      1888 2     Close the open help library.
: 1170      1889 2
: 1171      1890 2 INPUTS:
: 1172      1891 2
: 1173      1892 2     libindex =     address of longword containing index of library to close.
: 1174      1893 2
: 1175      1894 2 OUTPUTS:
: 1176      1895 2
: 1177      1896 2     None.
: 1178      1897 2
: 1179      1898 2 ROUTINE VALUE:
: 1180      1899 2
: 1181      1900 2     Always true.
: 1182      1901 2
: 1183      1902 2 !--
: 1184      1903 2
: 1185      1904 2 LOCAL
: 1186      1905 2     status;
: 1187      1906 2
: 1188      1907 3 IF NOT (status = lbr$close (.libindex))
: 1189      1908 2     THEN SIGNAL (.status);
: 1190      1909 2
: 1191      1910 2 RETURN true
: 1192      1911 2
: 1193      1912 1 END;
```

!Of close_library

		0000 00000 CLOSE_LIBRARY:				
		04	AC DD 00002	.WORD	Save nothing	: 1882
			01 FB 00005	PUSHL	LIBINDEX	: 1907
00000000G	00		50 E8 0000C	CALLS	#1, LBR\$CLOSE	
	09		50 DD 0000F	BLBS	STATUS, 1\$	
			01 FB 00011	PUSHL	STATUS	: 1908
00000000G	00		01 DO 00018 1\$:	CALLS	#1, LIB\$SIGNAL	
	50		04 0001B	MOVL	#1, R0	: 1910
				RET		: 1912

; Routine Size: 28 bytes, Routine Base: \$CODE\$ + 0984

```
: 1195      1913 1 %SBTTL 'Routine setup_keys';
: 1196      1914 1 ROUTINE setup_keys (getcddesc, keydescs, truekeys) =
: 1197      1915 2 BEGIN
: 1198      1916 2
: 1199      1917 2 ++
: 1200      1918 2 FUNCTIONAL DESCRIPTION:
: 1201      1919 2
: 1202      1920 2     This routine divides the descriptor for the line of keys into
: 1203      1921 2     individual descriptors for the individual keys.
: 1204      1922 2
: 1205      1923 2 INPUTS:
: 1206      1924 2
: 1207      1925 2    getcddesc =     address of descriptor for the set of keys currently being
: 1208      1926 2                      processed
: 1209      1927 2
: 1210      1928 2     keydescs =    address of vector of key descriptors to be returned
: 1211      1929 2
: 1212      1930 2     truekeys =    address of longword to contain the number of keys found
: 1213      1931 2
: 1214      1932 2 OUTPUTS:
: 1215      1933 2
: 1216      1934 2     keydescs, turekeys : as described above
: 1217      1935 2
: 1218      1936 2 ROUTINE VALUE:
: 1219      1937 2
: 1220      1938 2 --
: 1221      1939 2
: 1222      1940 2 MAP
: 1223      1941 2     truekeys : REF VECTOR [,BYTE],
: 1224      1942 2    getcddesc : REF BBLOCK,
: 1225      1943 2     keydescs : REF VECTOR [,BYTE];
: 1226      1944 2
: 1227      1945 2 LOCAL
: 1228      1946 2     paren,           ! Pointer to a left parenthesis
: 1229      1947 2     parm_begin,      ! Pointer to beginning of current key
: 1230      1948 2     parm_end,        ! Pointer to end of current key
: 1231      1949 2     next_qual,      ! Pointer to next qualifier
: 1232      1950 2     line_end;        ! Pointer to end of keys
: 1233      1951 2
: 1234      1952 2     paren = %ASCII '(';           ! Init paren ptr
: 1235      1953 2     CH$FILL (0, dsc$c_s_bln * hlp$c_maxkeys, .keydescs); ! Clear key descriptors
: 1236      1954 2
: 1237      1955 2
: 1238      1956 2     ! Convert command line to upper case. Then find the start of the help
: 1239      1957 2     ! keys, and fill in descriptors for them.
: 1240      1958 2
: 1241      1959 2
: 1242      1960 2     make_upper_case (.getcddesc, .getcddesc [dsc$a_pointer]); ! Convert command line to upper case
: 1243      1961 2     parm_end = .getcddesc [dsc$a_pointer]; ! Initialize pointers
: 1244      1962 2     line_end = .getcddesc [dsc$a_pointer] + .getcddesc [dsc$w_length];
: 1245      1963 2     next_qual = CH$FIND_CH (.getcddesc [dsc$w_length],
: 1246      1964 2                      .getcddesc [dsc$a_pointer], %ASCII '/');
: 1247      1965 2
: 1248      1966 2     INCRU level FROM 0 TO hlp$c_maxkeys - 1 ! Loop to search for keys
: 1249      1967 2     DO BEGIN
: 1250      1968 2
: 1251      1969 2         DO BEGIN ! Find first key that doesn't start
```



```
: 1252      1970  5      IF (parm_begin = CH$FIND_NOT_CH (.line_end - .parm_end,  
: 1253      1971  4      .parm_end, %ASCII ' ') EQL 0      ! Find start of key  
: 1254      1972  4      THEN parm_begin = .line_end;      ! Set right if none found  
: 1255      1973  4  
: 1256      1974  4      IF .next_qual NEQ 0      ! If there is a qualifier  
: 1257      1975  4      AND .parm_begin GEQU .next_qual      ! and qualifier is closer  
: 1258      1976  5      THEN BEGIN  
: 1259      1977  5      parm_begin = .next_qual;      ! then pick up the qualifier  
: 1260      1978  5      next_qual = CH$FIND_CH (.line_end - .parm_begin - 1,  
: 1261      1979  5      .parm_begin + 1, %ASCII '/');      ! and find the next one  
: 1262      1980  4      END;  
: 1263      1981  4  
: 1264      1982  5      IF (parm_end = CH$FIND_CH (.line_end - .parm_begin,  
: 1265      1983  4      .parm_begin, %ASCII ' ') EQL 0      ! Find end of key  
: 1266      1984  4      THEN parm_end = .line_end;  
: 1267      1985  4  
: 1268      1986  4      IF .next_qual NEQ 0      ! If a qualifier on line  
: 1269      1987  4      AND .parm_end GTRU .next_qual      ! and qualifier is closer  
: 1270      1988  4      THEN parm_end = .next_qual;      ! then it marks end of current param  
: 1271      1989  4  
: 1272      1990  4      END      ! Of until loop  
: 1273      1991  4  
: 1274      1992  4      UNTIL (.parm_end - .parm_begin EQL 0      ! No more keys  
: 1275      1993  3      OR CH$NEQ (1, .parm_begin, 1, paren));      ! or, key that starts with non-(  
: 1276      1994  3  
: 1277      1995  3      !  
: 1278      1996  3      Fill in string descriptor  
: 1279      1997  3      !  
: 1280      1998  3  
: 1281      1999  4      BEGIN  
: 1282      2000  4  
: 1283      2001  4      BIND  
: 1284      2002  4      curkeydesc = keydescs [dsc$c_s_bln * .level] : BBLOCK;      ! Descriptor for current key  
: 1285      2003  4  
: 1286      2004  4      IF (curkeydesc [dsc$w_length] = .parm_end - .parm_begin) EQL 0      ! If key length is zero  
: 1287      2005  5      THEN BEGIN  
: 1288      2006  5      IF .level EQL 0      ! And level agrees that really no mo  
: 1289      2007  5      THEN prompt_flags = .prompt_flags OR hcf$m_stay;      ! Then set prompt flag  
: 1290      2008  5      truekeys [0] = .level;      ! Set number of keys  
: 1291      2009  5      EXITLOOP;      ! And exit  
: 1292      2010  4      END;  
: 1293      2011  4      curkeydesc [dsc$a_pointer] = .parm_begin;      !Set pointer to start of key  
: 1294      2012  3      END;  
: 1295      2013  2      END;      !Of INCRU loop  
: 1296      2014  2  
: 1297      2015  2      RETURN true  
: 1298      2016  2  
: 1299      2017  1      END;      !Of setup_keys
```

00FC 00000 SETUP_KEYS:

0050	8F	00	57	28	DO 00002	.WORD	Save R2,R3,R4,R5,R6,R7	: 1914
			6E	00	2C 00005	MOVL	#40, PAREN	: 1952
						MOVCS	#0, (SP), #0, #80, @KEYDESCS	: 1953

		53	08 BC	D0 0000C	MOVL GETCMDDESC, R3	1960
			04 AC	DD 0000E	PUSHL 4(R3)	
			04 A3	DD 00012	PUSHL R3	
	0000V	CF	53 DD	00015	CALLS #2, MAKE_UPPER_CASE	
		55	02 FB	00017	MOVL 4(R3), PARM_END	1961
		52	A3 D0	0001C	(R3), LINE_END	1962
		52	63 3C	00020	ADDL2 4(R3), LINE_END	
04 B3		63	A3 C0	00023	LOCC #47, (R3), @4(R3)	1963
			2F 3A	00027	BNEQ 1\$	
			02 12	0002C	CLRL R1	
		56	51 D4	0002E	MOVL R1, NEXT_QUAL	
			51 D0	00030	CLRL LEVEL	1970
	50	52	54 D4	00033	SUBL3 PARM_END, LINE_END, RO	
	65	50	55 C3	00035	SKPC #32, -RO, (PARM_END)	
			20 3B	00039	BNEQ 3\$	
			02 12	0003D	CLRL R1	
			51 D4	0003F	MOVL R1, PARM_BEGIN	
		53	51 D0	00041	BNEQ 4\$	1971
			03 12	00044	MOVL LINE_END, PARM_BEGIN	1972
		53	52 D0	00046	TSTL NEXT_QUAL	1974
			56 D5	00049	BEQL 6\$	
			1A 13	0004B	CMP L PARM_BEGIN, NEXT_QUAL	1975
		56	53 D1	0004D	BLSSU 6\$	
			15 1F	00050	MOVL NEXT_QUAL, PARM_BEGIN	1977
	50	53	56 D0	00052	SUBL3 PARM_BEGIN, LINE_END, RO	1978
		52	53 C3	00055	DECL RO	
01 A3		50	50 D7	00059	LOCC #47, RO, 1(PARM_BEGIN)	
			2F 3A	0005B	BNEQ 5\$	
			02 12	00060	CLRL R1	
			51 D4	00062	MOVL R1, NEXT_QUAL	
		56	51 D0	00064	SUBL3 PARM_BEGIN, LINE_END, RO	1982
	50	52	53 C3	00067	LOCC #32, -RO, (PARM_BEGIN)	
	63	50	20 3A	0006B	BNEQ 7\$	
			02 12	0006F	CLRL R1	
			51 D4	00071	MOVL R1, PARM_END	
		55	51 D0	00073	BNEQ 8\$	1983
			03 12	00076	MOVL LINE_END, PARM_END	1984
		55	52 D0	00078	TSTL NEXT_QUAL	1986
			56 D5	0007B	BEQL 9\$	
			08 13	0007D	CMP L PARM_END, NEXT_QUAL	1987
		56	55 D1	0007F	BLEQU 9\$	
			03 1B	00082	MOVL NEXT_QUAL, PARM_END	1988
		55	56 D0	00084	CMP L PARM_END, PARM_BEGIN	1992
		53	55 D1	00087	BEQL 10\$	
			05 13	0008A	CMPB (PARM_BEGIN), PAREN	1993
		57	63 91	0008C	BEQL 2\$	
			A4 13	0008F	MOVAQ @KEYDESCS[LEVEL], RO	2002
		50	44 7E	00091	SUBL3 PARM_BEGIN, PARM_END, R1	2004
51		55	53 C3	00096	MOVW R1, (RO)	
		60	51 B0	0009A	TSTL R1	
			51 D5	0009D	BNEQ 12\$	
			0F 12	0009F	TSTL LEVEL	2006
			54 D5	000A1	BNEQ 11\$	
			05 12	000A3	BISB2 #2, PROMPT_FLAGS	2007
	0000' OC	CF	02 88	000A5	MOVB LEVEL, @TRUEKEYS	2008
		BC	54 90	000AA	BRB 13\$	2005
			0E 11	000AE		

LBR_OUTPUTHELP	Prompting and library searching help function	N 15	16-Sep-1984 02:04:00	VAX-11 Bliss-32 V4.0-742	Page 43
V04=000	Routine setup_keys		14-Sep-1984 12:37:45	[LBR.SRC]OUTPUTHLP.B32;1	(11)

04	A0	53	D0	000B0	12\$:	MOVL	PARM_BEGIN, 4(R0)	:	2011
		54	D6	000B4		INCL	LEVEL	:	1966
	09	54	D1	000B6		CMPL	LEVEL, #9	:	
		03	1A	000B9		BGTRU	13\$:	
		FF77	31	000BB		BRW	2\$:	
	50	01	D0	000BE	13\$:	MOVL	#1, R0	:	2015
		04	00	000C1		RET		:	2017

; Routine Size: 194 bytes, Routine Base: \$CODE\$ + 09A0

B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
[
\
]
^
_
`
a
b
c
d
e
f
g
h
i
j
k
l
m
n
o
p
q
r
s
t
u
v
w
x
y
z
{
|
}
~

```
: 1301      2018 1 %SBTTL 'Routine call_lbrhelp';
: 1302      2019 1 ROUTINE call_lbrhelp (helplibindex, outputwidth, printdata, keydescs) =
: 1303      2020 2 BEGIN
: 1304      2021 2
: 1305      2022 2 ++
: 1306      2023 2 FUNCTIONAL DESCRIPTION:
: 1307      2024 2
: 1308      2025 2     This routine calls the librarian function to extract help from
: 1309      2026 2     a particular help library.
: 1310      2027 2
: 1311      2028 2 INPUTS:
: 1312      2029 2
: 1313      2030 2     helplibindex = address of longword containing help library index
: 1314      2031 2
: 1315      2032 2     outputwidth = address of longword containing width of output line
: 1316      2033 2
: 1317      2034 2     printdata = address of data structure containg information for
: 1318      2035 2     the output driver
: 1319      2036 2
: 1320      2037 2     keydescs = address of vector of individual descriptors for each keyword
: 1321      2038 2
: 1322      2039 2 OUTPUTS:
: 1323      2040 2
: 1324      2041 2     None.
: 1325      2042 2
: 1326      2043 2 ROUTINE VALUE:
: 1327      2044 2
: 1328      2045 2     Status of call for help.
: 1329      2046 2
: 1330      2047 2 --
: 1331      2048 2
: 1332      2049 2 MAP
: 1333      2050 2     keydescs : REF VECTOR [,BYTE];
: 1334      2051 2
: 1335      2052 2 LOCAL
: 1336      2053 2     status: BBLOCK [LONG];
: 1337      2054 2
: 1338      2055 2 EXTERNAL
: 1339      2056 2     lbr$gl_control : REF BBLOCK;           ! Pointer to current library control block
: 1340      2057 2
: 1341      2058 2 BIND
: 1342      2059 2     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK;
: 1343      2060 2
: 1344      2061 2
: 1345      2062 2     Set this bit to make LBR$GET HELP handle the help on help case correctly.
: 1346      2063 2     This bit is not universally set for V2.0 compatibility reasons.
: 1347      2064 2
: 1348      2065 2 context [ctx$v_outputhlp] = true;
: 1349      2066 2 status = lbr$get_help (.helplibindex, .outputwidth,
: 1350      2067 2     output_driver, .printdata,
: 1351      2068 2     keydescs [dsc$c_s_bln * 0], keydescs [dsc$c_s_bln * 1],
: 1352      2069 2     keydescs [dsc$c_s_bln * 2], keydescs [dsc$c_s_bln * 3],
: 1353      2070 2     keydescs [dsc$c_s_bln * 4], keydescs [dsc$c_s_bln * 5],
: 1354      2071 2     keydescs [dsc$c_s_bln * 6], keydescs [dsc$c_s_bln * 7],
: 1355      2072 2     keydescs [dsc$c_s_bln * 8], keydescs [dsc$c_s_bln * 9]);
: 1356      2073 2 context [ctx$v_outputhlp] = false;
: 1357      2074 2
```


LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine call_lbrhelp

C 16
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 45
(12)

```
: 1358      2075  2 IF .status EQL lbr$_endtopic THEN      ! Abort help on this topic?
: 1359      2076      BEGIN
: 1360      2077      end_topic_flag = true;              ! Set flag to abort additional library listing and
: 1361      2078      status = true;                      ! change error to true
: 1362      2079      END;
: 1363      2080
: 1364      2081  2 RETURN .status
: 1365      2082  1 END;                                  ! Of call_lbrhelp
```

.EXTRN LBR\$GL_CONTROL

```
0004 00000 CALL_LBRHELP:
      50      0000G CF D0 00002      .WORD      Save R2
      52      0E      AO D0 00007      MOVL      LBR$GL_CONTROL, R0
05      A2      01 88 0000B      BISB2     #1, 5(R2)
      50      10      AC D0 0000F      MOVL      KEYDESCS, R0
      48      AO 9F 00013      PUSHAB     72(R0)
      40      AO 9F 00016      PUSHAB     64(R0)
      38      AO 9F 00019      PUSHAB     56(R0)
      30      AO 9F 0001C      PUSHAB     48(R0)
      28      AO 9F 0001F      PUSHAB     40(R0)
      20      AO 9F 00022      PUSHAB     32(R0)
      18      AO 9F 00025      PUSHAB     24(R0)
      10      AO 9F 00028      PUSHAB     16(R0)
      08      AO 9F 0002B      PUSHAB     8(R0)
      0C      50      DD 0002E      PUSHL     R0
      04      AC DD 00030      PUSHL     PRINTDATA
00000000G 7E      04      CF 9F 00033      PUSHAB     OUTPUT_DRIVER
05      A2      AC 7D 00037      MOVQ      HELPLIBINDEX, -(SP)
00000000G 8F      0E      FB 0003B      CALLS     #14, LBR$GET_HELP
      08      01 8A 00042      BICB2     #1, 5(R2)
      08      50      D1 00046      CMPL     STATUS, #LBR$_ENDTOPIC
      01      12 0004D      BNEQ      1$
      01      90 0004F      MOVB      #1, END_TOPIC_FLAG
      01      D0 00054      MOVL      #1, STATUS
      04      00057 1$:      RET
```

; Routine Size: 88 bytes, Routine Base: \$CODE\$ + 0A62


```
: 1367      2083 1 %SBTTL 'Routine output_driver';
: 1368      2084 1 ROUTINE output_driver (linedesc, helpflags, printdata, helplevel) =
: 1369      2085 2 BEGIN
: 1370      2086 2
: 1371      2087 2 !++
: 1372      2088 2 FUNCTIONAL DESCRIPTION:
: 1373      2089 2
: 1374      2090 2     Call user supplied output routine to print a line of help text
: 1375      2091 2     Also set various prompt flags and set initialize pointer user for
: 1376      2092 2     putting keys into subprompt buffer.
: 1377      2093 2
: 1378      2094 2 INPUTS:
: 1379      2095 2
: 1380      2096 2     linedesc =      address of descriptor for line to be output
: 1381      2097 2
: 1382      2098 2     helpflags =     address of flag longword describing contents of
: 1383      2099 2     text that is passed
: 1384      2100 2
: 1385      2101 2     printdata =    address of data structure containing flags, levels,
: 1386      2102 2     and other data for the output driver
: 1387      2103 2
: 1388      2104 2     helplevel =    address of longword containing the current key level
: 1389      2105 2
: 1390      2106 2 OUTPUTS:
: 1391      2107 2
: 1392      2108 2     The various data items in printdata are updated and the user supplied
: 1393      2109 2     output routine is called to output the linedesc.
: 1394      2110 2
: 1395      2111 2 ROUTINE VALUE:
: 1396      2112 2
: 1397      2113 2     False, if call to user supplied routine returns false.
: 1398      2114 2     True, otherwise.
: 1399      2115 2 --
: 1400      2116 2
: 1401      2117 2 MAP
: 1402      2118 2     linedesc : REF BBLOCK,
: 1403      2119 2     printdata : REF BBLOCK;
: 1404      2120 2
: 1405      2121 2 BIND
: 1406      2122 2     true_keys = printdata [hpd$b truekeys] : SIGNED BYTE,      ! Number of help keys
: 1407      2123 2     help_level = printdata [hpd$b helplevel] : BYTE,          ! Current key depth
: 1408      2124 2     print_flags = printdata [hpd$b printflag] : BBLOCK,      ! Flags for output driver
: 1409      2125 2     add_info_level = printdata [hpd$l_subpmtlev],              ! Current prompt level
: 1410      2126 2     sub_prompt_ptr = printdata [hpd$l_subpmtptr],              ! Ptr used for filling sub-prompt buffer
: 1411      2127 2     length_array = printdata [hpd$l_lenarray] : REF VECTOR,    ! Address of key length array
: 1412      2128 2     output_routine = printdata [hpd$l_outputrou];              ! User specified output routine
: 1413      2129 2
: 1414      2130 2 OWN
: 1415      2131 2     topics_available : COUNTEDSTRING ('Information available:');
: 1416      2132 2
: 1417      2133 2 LOCAL
: 1418      2134 2     keylevel,          ! Local equivalent of parameter helplevel
: 1419      2135 2     flags,             ! Local equivalent of parameter helpflags
: 1420      2136 2     output_buf : BBLOCK [hlp$c_pagesize],      ! Local buffer to store line to be output
: 1421      2137 2     output_desc : BBLOCK [dsc$c_s_bln],         ! Local output descriptor
: 1422      2138 2     ptr,              ! Pointer into output buffer
: 1423      2139 2     spaces : WORD,    ! Number of spaces to indent output line
```



```

: 1424      2140 2      status;
: 1425      2141 2
: 1426      2142 2      status = true;
: 1427      2143 2      flags = ..helpflags;
: 1428      2144 2
: 1429      2145 2      IF NOT .print_flags [hpd$y_all] AND
: 1430      2146 2      ((.flags AND hlp$m_nohlptxt) NEQ 0)
: 1431      2147 2      THEN BEGIN
: 1432      2148 2          print_flags [hpd$y_found] = false;
: 1433      2149 2          RETURN true;
: 1434      2150 2      END;
: 1435      2151 2
: 1436      2152 2      print_flags [hpd$y_found] = true;
: 1437      2153 2      print_flags [hpd$y_all] = true;
: 1438      2154 2
: 1439      2155 2      keylevel = ..helplevel;
: 1440      2156 2      output_desc = 0;
: 1441      2157 2      output_desc [dsc$w_length] = .linedesc [dsc$w_length];
: 1442      2158 2      output_desc [dsc$a_pointer] = output_buf;
: 1443      2159 2
: 1444      2160 2      IF .flags EQL 0
: 1445      2161 2      THEN prompt_flags = .prompt_flags OR hcf$m_info;
: 1446      2162 2
: 1447      2163 2      !
: 1448      2164 2      ! If processing key name and moving down a prompt level,
: 1449      2165 2      ! then build subtopic prompt
: 1450      2166 2      !
: 1451      2167 2
: 1452      2168 2      IF ((.flags AND hlp$m_keynamlin) NEQ 0) AND
: 1453      2169 2      ((.prompt_flags AND hcf$m_stay) EQL 0)
: 1454      2170 2      THEN BEGIN
: 1455      2171 2          IF (.prompt_flags AND hcf$m_info) NEQ 0
: 1456      2172 2          THEN prompt_flags = .prompt_flags OR hcf$m_stay
: 1457      2173 2          ELSE BEGIN
: 1458      2174 2              IF (.keylevel NEQ .help_level) AND (.output_desc [dsc$w_length] NEQ 0)
: 1459      2175 2              THEN BEGIN
: 1460      2176 2                  help_level = .keylevel;
: 1461      2177 2                  CHSMOVE (.output_desc [dsc$w_length], .linedesc [dsc$a_pointer],
: 1462      2178 2                      .sub_prompt_ptr);
: 1463      2179 2                  length_array [.help_level] = .output_desc [dsc$w_length];
: 1464      2180 2                  sub_prompt_ptr = .sub_prompt_ptr + .output_desc [dsc$w_length];
: 1465      2181 2                  (.sub_prompt_ptr)<0,85 = 32;
: 1466      2182 2                  sub_prompt_ptr = .sub_prompt_ptr + 1;
: 1467      2183 2              END;
: 1468      2184 2          END;
: 1469      2185 2          add_info_level = .keylevel;
: 1470      2186 2      END;
: 1471      2187 2
: 1472      2188 2      !
: 1473      2189 2      ! If no help text found, then stay at current level and signal that
: 1474      2190 2      ! help was not found.
: 1475      2191 2      !
: 1476      2192 2
: 1477      2193 2      IF (.flags AND hlp$m_nohlptxt) NEQ 0
: 1478      2194 2      THEN BEGIN
: 1479      2195 2          prompt_flags = .prompt_flags OR hcf$m_stay;
: 1480      2196 2          IF (.print_flags [hpd$y_init] EQL 0)

```

! Status if user output routine never called
! Copy helpflags locally

! If not printing all help text
! and no help text found

! Then set no help found flag
! And return

! Else say help was found
! Keyword was found, so this will ensure tha
! will not be search, even if no text follo
! Copy key level locally
! Init local descriptor for output line

! If help info found
! Then say so


```
: 1481      2197 3      THEN print_flags [hpd$v_init] = 1;
: 1482      2198 2      END;
: 1483      2199 2
: 1484      2200 2      :
: 1485      2201 2      : If additional info to be found, say so
: 1486      2202 2      :
: 1487      2203 2      :
: 1488      2204 2      IF (.flags AND hlp$m_otherinfo) NEQ 0
: 1489      2205 2      THEN prompt_flags = .prompt_flags OR hcf$m_more;
: 1490      2206 2
: 1491      2207 2      :
: 1492      2208 2      : Format output line and then call user supplied output routine.
: 1493      2209 2      :
: 1494      2210 2
: 1495      2211 3      IF ((.true_keys NEQ 0) OR                                ! Not at first topic level
: 1496      2212 3      .help_flags [hlp$v_help] OR                            ! Want help on help with topic list
: 1497      2213 3      ((.flags AND hlp$m_otherinfo) NEQ 0))                    ! List of additional info
: 1498      2214 3      THEN IF (.keylevel GTR 0) OR
: 1499      2215 3      ((.flags AND hlp$m_keynamlin) EQL 0)
: 1500      2216 3      THEN BEGIN
: 1501      2217 3      IF .linedesc [dsc$w_length] NEQ 0
: 1502      2218 4      THEN BEGIN
: 1503      2219 4
: 1504      2220 5      IF ((.flags AND hlp$m_keynamlin) NEQ 0)
: 1505      2221 4      THEN spaces = (.keylevel - 1) * hlp$c_indent
: 1506      2222 5      ELSE IF ((.flags AND hlp$m_nohlptxt) NEQ 0)
: 1507      2223 5      AND (.keylevel EQL 0)
: 1508      2224 4      THEN spaces = hlp$c_indent
: 1509      2225 4      ELSE spaces = .keylevel * hlp$c_indent;
: 1510      2226 4
: 1511      2227 4      ptr = CH$FILL (%ASCII ' ', .spaces, output_buf);
: 1512      2228 5      IF ((.true_keys EQL 0) AND NOT .help_flags [hlp$v_help])
: 1513      2229 5      THEN BEGIN
: 1514      2230 5      true_keys = -1;
: 1515      2231 5      output_desc [dsc$w_length] = .topics_available [0];
: 1516      2232 5      CH$MOVE (.output_desc [dsc$w_length], topics_available + 1, .ptr);
: 1517      2233 5      END
: 1518      2234 5      ELSE BEGIN
: 1519      2235 5      CH$MOVE (.output_desc [dsc$w_length], .linedesc [dsc$a_pointer], .ptr);
: 1520      2236 4      END;
: 1521      2237 4      output_desc [dsc$w_length] = .spaces + .output_desc [dsc$w_length];
: 1522      2238 3      END;
: 1523      2239 3
: 1524      2240 3      status = (.output_routine) (output_desc);
: 1525      2241 3
: 1526      2242 2      END;
: 1527      2243 2
: 1528      2244 2      RETURN .status;
: 1529      2245 1      END;

!Of output_driver
```

.PSECT \$OWNS,NOEXE,2

16 00054 TOPICS_AVAILABLE:

.BYTE 22

61 76 61 20 6E 6F 69 74 61 6D 72 6F 66 6E 49 00055 .ASCII \Information available:\

:

3A 65 6C 62 61 6C 69 00064

:

.PSECT \$CODE\$,NOWRT,2

				OFFC	00000	OUTPUT_DRIVER:				
			5E	FDF8	CE	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	2084
			57	0C	AC	D0	00007	MOVAB	-520(SP), SP	2122
			59	12	A7	9E	0000B	MOVL	PRINTDATA, R7	2124
					01	D0	0000F	MOVAB	18(R7), R9	2142
			5A	08	BC	D0	00011	PUSHL	#1	2143
0A			69		01	E0	00015	MOVL	@HELPFLAGS, FLAGS	2145
			07		5A	E9	00019	BBS	#1, (R9) 1\$	2146
			69		04	8A	0001C	BLBC	FLAGS, 1\$	2148
			50		01	D0	0001F	BICB2	#4, (R9)	2149
							04	MOVL	#1, R0	
			69		06	88	00023	RET		
			56		BC	D0	00026	BISB2	#6, (R9)	2153
				10	AE	D4	0002A	MOVL	@HELPLEVEL, KEYLEVEL	2155
			58	04	AC	D0	0002D	CLRL	OUTPUT_DESC	2156
			AE	04	68	B0	00031	MOVL	LINEDESC, R8	2157
	04		AE		AE	9E	00035	MOVW	(R8), OUTPUT_DESC	
	08		AE	0C	5A	D5	0003A	MOVAB	OUTPUT_BUF, OUTPUT_DESC+4	2158
					05	12	0003C	TSTL	FLAGS	2160
		0000'	CF		08	88	0003E	BNEQ	2\$	
					5B	D4	00043	BISB2	#8, PROMPT_FLAGS	2161
	48		5A		01	E1	00045	CLRL	R11	2168
					5B	D6	00049	BBC	#1, FLAGS, 5\$	
	40		CF		01	E0	0004B	INCL	R11	
	07	0000'	CF		03	E1	00051	BBS	#1, PROMPT_FLAGS, 5\$	2169
		0000'	CF		02	88	00057	BBC	#3, PROMPT_FLAGS, 3\$	2171
		0000'	CF		02	88	00057	BISB2	#2, PROMPT_FLAGS	2172
					2F	11	0005C	BRB	4\$	
56	11	A7	08		00	ED	0005E	CMPZV	#0, #8, 17(R7), KEYLEVEL	2174
					27	13	00064	BEQL	4\$	
				04	AE	B5	00066	TSTW	OUTPUT_DESC	
					22	13	00069	BEQL	4\$	
		11	A7		56	90	0006B	MOVB	KEYLEVEL, 17(R7)	2176
	00	B7	04	B8	04	AE	28	MOVAB	OUTPUT_DESC, @4(R8), @0(R7)	2178
			50		11	A7	9A	MOVZBL	17(R7), R0	2179
			OC	B740	04	AE	3C	MOVZWL	OUTPUT_DESC, @12(R7)[R0]	
			50		04	AE	3C	MOVZWL	OUTPUT_DESC, R0	2180
			67		50	C0	00084	ADDL2	R0, (R7)	
		00	B7		20	90	00087	MOVB	#32, @0(R7)	2181
					67	D6	0008B	INCL	(R7)	2182
		04	A7		56	D0	0008D	MOVL	KEYLEVEL, 4(R7)	2185
					51	D4	00091	CLRL	R1	2193
			0D		5A	E9	00093	BLBC	FLAGS, 6\$	
					51	D6	00096	INCL	R1	
		0000'	CF		02	88	00098	BISB2	#2, PROMPT_FLAGS	2195
			03		69	E8	0009D	BLBS	(R9), 6\$	2196
			69		01	88	000A0	BISB2	#1, (R9)	2197
					50	D4	000A3	CLRL	R0	2204
	07		5A		02	E1	000A5	BBC	#2, FLAGS, 7\$	
					50	D6	000A9	INCL	R0	
		0000'	CF		04	88	000AB	BISB2	#4, PROMPT_FLAGS	2205

			10	A7	95	000B0	7\$:	TSTB	16(R7)	: 2211
				09	12	000B3		BNEQ	8\$: 2212
03	0000'	CF		05	E0	000B5		BBS	#5, HELP_FLAGS, 8\$: 2213
		62		50	E9	000BB		BLBC	R0, 16\$: 2214
				56	D5	000BE	8\$:	TSTL	KEYLEVEL	: 2215
				04	14	000C0		BGTR	9\$: 2217
5A		5A		01	E0	000C2		BBS	#1, FLAGS, 16\$: 2220
				68	B5	000C6	9\$:	TSTW	(R8)	: 2221
				4C	13	000C8		BEQL	15\$: 2222
		0A		5B	E9	000CA		BLBC	R11, 10\$: 2223
		50	FF	A6	9E	000CD		MOVAB	-1(R6), R0	: 2224
5A		50		02	A5	000D1		MULW3	#2, R0, SPACES	: 2225
				10	11	000D5		BRB	12\$: 2227
		09		51	E9	000D7	10\$:	BLBC	R1, 11\$: 2228
				56	D5	000DA		TSTL	KEYLEVEL	: 2230
				05	12	000DC		BNEQ	11\$: 2231
		5A		02	B0	000DE		MOVW	#2, SPACES	: 2232
				04	11	000E1		BRB	12\$: 2233
	5A	56		02	A5	000E3	11\$:	MULW3	#2, KEYLEVEL, SPACES	: 2234
5A	20	6E		00	2C	000E7	12\$:	MOVCS	#0, (SP), #32, SPACES, OUTPUT_BUF	: 2235
			0C	AE		000EC				: 2236
			10	A7	95	000EE		TSTB	16(R7)	: 2237
				19	12	000F1		BNEQ	13\$: 2238
13	0000'	CF		05	E0	000F3		BBS	#5, HELP_FLAGS, 13\$: 2239
	10	A7		01	8E	000F9		MNEGB	#1, 16(R7)	: 2240
	04	AE	0000'	CF	9B	000FD		MOVZBW	TOPICS_AVAILABLE, OUTPUT_DESC	: 2241
63	0000'	CF	04	AE	23	00103		MOVCS	OUTPUT_DESC, TOPICS_AVAILABLE+1, (PTR)	: 2242
				06	11	0010A		BRB	14\$: 2243
63	04	B8	04	AE	28	0010C	13\$:	MOVCS	OUTPUT_DESC, @4(R8), (PTR)	: 2244
	04	AE		5A	A0	00112	14\$:	ADDW2	SPACES, OUTPUT_DESC	: 2245
			04	AE	9F	00116	15\$:	PUSHAB	OUTPUT_DESC	: 2246
	08	B7		01	FB	00119		CALLS	#1, @8(R7)	: 2247
		6E		50	D0	0011D		MOVL	R0, STATUS	: 2248
		50		6E	D0	00120	16\$:	MOVL	STATUS, R0	: 2249
				04		00123		RET		: 2250

; Routine Size: 292 bytes, Routine Base: \$CODE\$ + 0ABA


```
: 1531      2246 1 %SBTTL 'Routine libs_available';
: 1532      2247 1 ROUTINE libs_available (outputroutine, outputwidth) =
: 1533      2248 2 BEGIN
: 1534      2249 2
: 1535      2250 2 !++
: 1536      2251 2 FUNCTIONAL DESCRIPTION
: 1537      2252 2
: 1538      2253 2     Output a list of the define default libraries.
: 1539      2254 2
: 1540      2255 2 INPUTS:
: 1541      2256 2
: 1542      2257 2     outputroutine = address of the user supplied output routine
: 1543      2258 2
: 1544      2259 2     outputwidth = address of a longword containing the output line width
: 1545      2260 2
: 1546      2261 2 OUTPUTS:
: 1547      2262 2
: 1548      2263 2     The list is output by using the user supplied routine.
: 1549      2264 2
: 1550      2265 2 ROUTINE VALUE:
: 1551      2266 2
: 1552      2267 2     Always true.
: 1553      2268 2 !--
: 1554      2269 2
: 1555      2270 2 LOCAL
: 1556      2271 2     acmode,
: 1557      2272 2     blank_line : BBLOCK [dsc$c_s_bln],
: 1558      2273 2     filespec : BBLOCK [dsc$c_s_bln],
: 1559      2274 2     filespec_buf : BBLOCK [nam$c_maxrss],
: 1560      2275 2     filename : BBLOCK [dsc$c_s_bln],
: 1561      2276 2     filename_buf : VECTOR [filename_length, BYTE],
: 1562      2277 2     libno,
: 1563      2278 2     output_desc : BBLOCK [dsc$c_s_bln],
: 1564      2279 2     output_buf : BBLOCK [hlp$c_pagesize],
: 1565      2280 2     status;
: 1566      2281 2
: 1567      2282 2 OWN
: 1568      2283 2     header : COUNTEDSTRING (' Additional help libraries available (type @name for topics):');
: 1569      2284 2
: 1570      2285 2     filespec = 0;                                ! Init file spec desc
: 1571      2286 2     filespec [dsc$a_pointer] = filespec_buf;
: 1572      2287 2
: 1573      2288 2     libno = -1;                                ! Init search
: 1574      2289 2     status = false;                                ! Init status
: 1575      2290 2
: 1576      2291 2     WHILE NOT .status                                ! Loop searching for a library
: 1577      2292 2     DO BEGIN
: 1578      2293 2         IF NOT tran_next_lib (filespec, acmode, libno) ! Get first lib spec
: 1579      2294 2         THEN RETURN true;                            ! If none, then return
: 1580      2295 2         status = file_present (filespec);                ! Is file present?
: 1581      2296 2     END;
: 1582      2297 2
: 1583      2298 2     filename = 0;                                ! Init file name desc
: 1584      2299 2     filename [dsc$a_pointer] = filename_buf;
: 1585      2300 2
: 1586      2301 2     blank_line = 0;                                ! Init blank line desc
: 1587      2302 2     blank_line [dsc$a_pointer] = output_buf;
```

```

1588      2303 2
1589      2304 2 output_desc = .header [0];                                ! Init header desc
1590      2305 2 output_desc [dsc$a_pointer] = header [1];
1591      2306 2
1592      2307 2 (.outputroutine) (blank_line);                                ! Output header
1593      2308 2 (.outputroutine) (output_desc);
1594      2309 2 (.outputroutine) (blank_line);
1595      2310 2
1596      2311 2 output_desc = 0;                                            ! Reuser output desc for
1597      2312 2 output_desc [dsc$a_pointer] = output_buf;                  ! lib name lists
1598      2313 2 CH$FILC ('X'20', ..outputwidth, output_buf);
1599      2314 2
1600      2315 2 switch_libname (filespec, filename);                          ! Get first lib name
1601      2316 2 output_desc [dsc$w_length] = .filename[dsc$w_length] + 2;  ! Move it into
1602      2317 2 CH$MOVE (.filename[dsc$w_length], filename_buf, output_buf + 2); ! the list.
1603      2318 2
1604      2319 2 WHILE tran_next_lib(filespec, acmode, libno) DO              ! While more libs translate,
1605      2320 2     IF file_present(filespec) THEN                          ! if the file exists,
1606      2321 2         BEGIN
1607      2322 2             switch_libname(filespec, filename);              ! get its name.
1608      2323 2
1609      2324 2             ! If we can't fit a couple of spaces and the filename on the
1610      2325 2             ! line we're working on, send the line to the output routine and
1611      2326 2             ! start a new one:
1612      2327 2
1613      2328 2             IF (.output_desc[dsc$w_length] + 2 + .filename[dsc$w_length]) GTRU ..outputwidth THEN
1614      2329 2                 BEGIN
1615      2330 2                     (.outputroutine) (output_desc);
1616      2331 2                     output_desc[dsc$w_length] = 0;
1617      2332 2                     CH$FILC ('X'20', ..outputwidth, output_buf);
1618      2333 2                 END;
1619      2334 2
1620      2335 2             ! Move the filename onto the line (leaving two padding spaces):
1621      2336 2             !
1622      2337 2             CH$MOVE (.filename[dsc$w_length], filename_buf,
1623      2338 2                 output_buf + .output_desc[dsc$w_length] + 2);
1624      2339 2             output_desc[dsc$w_length] = .output_desc[dsc$w_length] + 2 + .filename[dsc$w_length];
1625      2340 2             END;
1626      2341 2
1627      2342 2 (.outputroutine) (output_desc);                                ! Output last line built
1628      2343 2 RETURN true;
1629      2344 1 END;

```

```

65 68 20 6C 61 6E 6F 69 74 69 64 64 41 20 3E 0006B
76 61 20 73 65 69 72 61 72 62 69 6C 20 70 6C 0006C
                                0006D HEADER: .BLKB 1
                                0007C .BYTE 62
74 20 72 6F 66 20 65 6D 61 6E 40 20 65 70 79 0008B
                                00095 .ASCII \ Additional help libraries available (t\
                                000A4 .ASCII \type @name for topics):\
                                6F

```


.PSECT \$CODE\$,NOWRT,2

03FC 00000 LIBS_AVAILABLE:

	59	FC2B	CF	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9	2247
	5E	FCB4	CE	9E	00007	MOVAB	TRAN_NEXT_LIB, R9	
		F0	AD	D4	0000C	MOVAB	-8447(SP), -SP	
F4	AD	FEF0	CD	9E	0000F	CLRL	FILESPEC	2285
	7E		01	CE	00015	MOVAB	FILESPEC BUF, FILESPEC+4	2286
			52	D4	00018	MNEGL	#1, LIBNO	2288
	1E		52	E8	0001A	CLRL	STATUS	2289
			5E	DD	0001D	BLBS	STATUS, 3\$	2291
		08	AE	9F	0001F	PUSHL	SP	2293
		F0	AD	9F	00022	PUSHAB	ACMODE	
	69		03	FB	00025	PUSHAB	FILESPEC	
	03		50	E8	00028	CALLS	#3, TRAN_NEXT_LIB	
			00DF	31	0002B	BLBS	R0, 2\$	
		F0	AD	9F	0002E	BRW	7\$	
0000V	CF		01	FB	00031	PUSHAB	FILESPEC	2295
	52		50	D0	00036	CALLS	#1, FILE_PRESENT	
			DF	11	00039	MOVL	R0, STATUS	
		FEE8	CD	D4	0003B	BRB	1\$	2291
FEEC	CD	FEC0	CD	9E	0003F	CLRL	FILENAME	2298
		F8	AD	D4	00046	MOVAB	FILENAME BUF, FILENAME+4	2299
FC	AD	08	AE	9E	00049	CLRL	BLANK LINE	2301
FEB8	CD	0000	CF	9A	0004E	MOVAB	OUTPUT_BUF, BLANK LINE+4	2302
FEB8	CD	0000	CF	9E	00055	MOVZBL	HEADER, OUTPUT_DESC	2304
	58	04	AC	D0	0005C	MOVAB	HEADER+1, OUTPUT_DESC+4	2305
		F8	AD	9F	00060	MOVL	OUTPUTROUTINE, R8	2307
	68		01	FB	00063	PUSHAB	BLANK LINE	
		FEB8	CD	9F	00066	CALLS	#1, (R8)	
	68		01	FB	0006A	PUSHAB	OUTPUT_DESC	2308
		F8	AD	9F	0006D	CALLS	#1, (R8)	
	68		01	FB	00070	PUSHAB	BLANK LINE	2309
		FEB8	CD	D4	00073	CALLS	#1, (R8)	
		08	AE	9E	00077	CLRL	OUTPUT_DESC	2311
08	BC	20	00	2C	0007D	MOVAB	OUTPUT_BUF, OUTPUT_DESC+4	2312
			08	AE	00083	MOVCS	#0, (SP), #32, @OUTPUTWIDTH, OUTPUT_BUF	2313
		FEE8	CD	9F	00085	PUSHAB	FILENAME	2315
		F0	AD	9F	00089	PUSHAB	FILESPEC	
	85	A9	02	FB	0008C	CALLS	#2, SWITCH_LIBNAME	
FEB8	CD	FEE8	02	A1	00090	ADDW3	#2, FILENAME, OUTPUT_DESC	2316
OA	AE	FEC0	CD	28	00098	MOVCS	FILENAME, FILENAME_BUF, OUTPUT_BUF+2	2317
			5E	DD	000A1	PUSHL	SP	2319
		08	AE	9F	000A3	PUSHAB	ACMODE	
		F0	AD	9F	000A6	PUSHAB	FILESPEC	
	69		03	FB	000A9	CALLS	#3, TRAN_NEXT_LIB	
	57		50	E9	000AC	BLBC	R0, 6\$	
		F0	AD	9F	000AF	PUSHAB	FILESPEC	2320
0000V	CF		01	FB	000B2	CALLS	#1, FILE_PRESENT	
	E7		50	E9	000B7	BLBC	R0, 4\$	
		FEE8	CD	9F	000BA	PUSHAB	FILENAME	2322
		F0	AD	9F	000BE	PUSHAB	FILESPEC	
85	A9		02	FB	000C1	CALLS	#2, SWITCH_LIBNAME	
	50	FEB8	CD	3C	000C5	MOVZWL	OUTPUT_DESC, R0	2328
	57	FEE8	CD	3C	000CA	MOVZWL	FILENAME, R7	
	50	02	A740	9E	000CF	MOVAB	2(R7)(R0), R0	

LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine libs_available

L 16
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 54
(14)

08	BC		50	D1	000D4		CMPL	R0, @OUTPUTWIDTH	:	
			13	1B	000D8		BLEQU	5\$:	
		FEB8	CD	9F	000DA		PUSHAB	OUTPUT_DESC	:	2330
	68		01	FB	000DE		CALLS	#1, (R8)	:	
		FEB8	CD	B4	000E1		CLRW	OUTPUT_DESC	:	2331
08	BC	20	6E	00	2C	000E5	MOVC5	#0, (SP), #32, @OUTPUTWIDTH, OUTPUT_BUF	:	2332
			08	AE		000EB			:	
	0A AE46	FEC0	56	FEB8	CD	3C	000ED	5\$: MOVZWL	OUTPUT_DESC, R6	2338
			CD	57	28	000F2	MOVC3	R7, FILENAME_BUF, OUTPUT_BUF+2[R6]	:	
			50	02	A746	9E	000FA	MOVAB	2(R7)[R6], R0	2339
		FEB8	CD	50	B0	000FF	MOVW	R0, OUTPUT_DESC	:	
				9B	11	00104	BRB	4\$:	2320
			68	FEB8	CD	9F	00106	6\$: PUSHAB	OUTPUT_DESC	2342
			50		01	FB	0010A	CALLS	#1, (R8)	:
					01	D0	0010D	7\$: MOVL	#1, R0	2343
					04	00110	RET		:	2344

; Routine Size: 273 bytes, Routine Base: \$CODE\$ + 0BDE


```
: 1631      2345 1 %SBTTL 'Routine file_present';
: 1632      2346 1 ROUTINE file_present(filename) =
: 1633      2347 2 BEGIN
: 1634      2348 2
: 1635      2349 2 |++
: 1636      2350 2 | FUNCTIONAL DESCRIPTION:
: 1637      2351 2 |
: 1638      2352 2 |     Return success if the file exists.
: 1639      2353 2 |
: 1640      2354 2 | INPUTS:
: 1641      2355 2 |
: 1642      2356 2 |     filename = address of desc of file name
: 1643      2357 2 |
: 1644      2358 2 | OUTPUTS:
: 1645      2359 2 |
: 1646      2360 2 |     None.
: 1647      2361 2 |
: 1648      2362 2 | ROUTINE VALUE:
: 1649      2363 2 |
: 1650      2364 2 |     True, if file exists
: 1651      2365 2 |     False, if it doesn't
: 1652      2366 2 |
: 1653      2367 2 | --
: 1654      2368 2 |
: 1655      2369 2 MAP
: 1656      2370 2     filename : REF BBLOCK;
: 1657      2371 2
: 1658      2372 2 LOCAL
: 1659      2373 2     fab : BBLOCK [fab$c_bln],
: 1660      2374 2     nam : BBLOCK [nam$c_bln],
: 1661      2375 2     string : BBLOCK [nam$c_maxrss],
: 1662      2376 2     status;
: 1663      2377 2
: 1664      P 2378 2 $NAM_INIT ( NAM = nam,
: 1665      P 2379 2     ESS = nam$c_maxrss,
: 1666      2380 2     ESA = string);
: 1667      2381 2
: 1668      P 2382 2 $FAB_INIT ( FAB = fab,
: 1669      P 2383 2     FNS = .filename [dsc$w_length],
: 1670      P 2384 2     FNA = .filename [dsc$a_pointer],
: 1671      P 2385 2     DNS = .syshelp [0],
: 1672      P 2386 2     DNA = syshelp [1],
: 1673      2387 2     NAM = nam);
: 1674      2388 2
: 1675      2389 3 IF (status = $PARSE (FAB = fab))
: 1676      2390 2     THEN (status = $SEARCH (FAB = fab));
: 1677      2391 2
: 1678      2392 2 RETURN .status;
: 1679      2393 1 END;
```

.EXTRN SYS\$SEARCH

003C 00000 FILE_PRESENT:

5E

FE50

CE 9E 00002

WORD
MOVABSave R2,R3,R4,R5
-432(SP), SP: 2346
:

LBR_OUTPUTHELP		Prompting and library searching help function		B 1		16-Sep-1984 02:04:00		VAX-11 Bliss-32 V4.0-742		Page 56	
V04=000		Routine file_present				14-Sep-1984 12:37:45		[LBR.SRC]OUTPUTHELP.B32;1		(15)	

0060	8F	00	6E	FF50	00	2C	00007	MOVCS	#0, (SP), #0, #96, \$RMS_PTR	:	2380	
				CD	CD		C000E			:		
				6002	8F	B0	00011	MOVW	#24578, \$RMS_PTR	:		
					01	8E	00018	MNEGB	#1, \$RMS_PTR+10	:		
					6E	9E	0001D	MOVAB	STRING, \$RMS_PTR+12	:		
0050	8F	00	6E		00	2C	00022	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	:	2387	
					B0	AD	00029			:		
					5003	8F	B0	0002B	MOVW	#20483, \$RMS_PTR	:	
						02	90	00031	MOVB	#2, \$RMS_PTR+22	:	
						02	90	00035	MOVB	#2, \$RMS_PTR+31	:	
						CD	9E	00039	MOVAB	NAM, \$RMS_PTR+40	:	
						AC	D0	0003F	MOVL	FILENAME, -R0	:	
						A0	D0	00043	MOVL	4(R0), \$RMS_PTR+44	:	
						CF	9E	00048	MOVAB	SYSHELP+1, \$RMS_PTR+48	:	
						60	90	0004E	MOVB	(R0), \$RMS_PTR+52	:	
						CF	90	00052	MOVB	SYSHELP, \$RMS_PTR+53	:	
						AD	9F	00058	PUSHAB	FAB	:	2389
						01	FB	0005B	CALLS	#1, SYSSPARSE	:	
						50	E9	00062	BLBC	STATUS, 1\$:	
						AD	9F	00065	PUSHAB	FAB	:	2390
						01	FB	00068	CALLS	#1, SYSSSEARCH	:	
						04	0006F	1\$:	RET	:	2393	

; Routine Size: 112 bytes, Routine Base: \$CODE\$ + 0CEF


```
: 1681      2394 1 %SBTTL 'Routine nohelp_log';
: 1682      2395 1 ROUTINE nohelp_log (logdesc) =
: 1683      2396 2 BEGIN
: 1684      2397 2
: 1685      2398 2 |++
: 1686      2399 2 | FUNCTIONAL DESCRIPTION:
: 1687      2400 2 |
: 1688      2401 2 |     If the logical name for a log file is defined, then put a record
: 1689      2402 2 |     into the specified log file. If that file does not already exist,
: 1690      2403 2 |     then create it.
: 1691      2404 2 |
: 1692      2405 2 | INPUTS:
: 1693      2406 2 |
: 1694      2407 2 |     logdesc = address of string descriptor for record to be output
: 1695      2408 2 |
: 1696      2409 2 | OUTPUTS:
: 1697      2410 2 |
: 1698      2411 2 |     None.
: 1699      2412 2 |
: 1700      2413 2 | ROUTINE VALUE:
: 1701      2414 2 |
: 1702      2415 2 |     Always true.
: 1703      2416 2 |
: 1704      2417 2 | --
: 1705      2418 2 |
: 1706      2419 2 | MAP
: 1707      2420 2 |     logdesc : REF BBLOCK;
: 1708      2421 2 |
: 1709      2422 2 | LOCAL
: 1710      2423 2 |     logresult : VECTOR [nam$c_maxrss, BYTE],
: 1711      2424 2 |     logrsdesc : BBLOCK [dsc$c_s_bln],
: 1712      2425 2 |     lognam    : BBLOCK [nam$c_bln],
: 1713      2426 2 |     logfab    : BBLOCK [fab$c_bln],
: 1714      2427 2 |     lograb    : BBLOCK [rab$c_bln],
: 1715      2428 2 |     logfile   : BBLOCK [dsc$c_s_bln],
: 1716      2429 2 |     logfiletrn : BBLOCK [dsc$c_s_bln],
: 1717      2430 2 |     status;
: 1718      2431 2 |
: 1719      2432 2 | OWN
: 1720      2433 2 |     logstring : COUNTEDSTRING ('HELP$LOG');
: 1721      2434 2 |
: 1722      2435 2 |     logfile [dsc$w_length] = .logstring [0];
: 1723      2436 2 |     logfile [dsc$a_pointer] = logstring [1];
: 1724      2437 2 |
: 1725      2438 2 |     logfiletrn [dsc$w_length] = nam$c_maxrss;
: 1726      2439 2 |     logfiletrn [dsc$a_pointer] = logresult;
: 1727      2440 2 |
: 1728      P 2441 2 | $NAM_INIT ( NAM = lognam,
: 1729      P 2442 2 |             ESS = nam$c_maxrss,
: 1730      P 2443 2 |             ESA = logresult,
: 1731      P 2444 2 |             RSS = nam$c_maxrss,
: 1732      2445 2 |             RSA = logresult);
: 1733      2446 2 |
: 1734      P 2447 2 | $FAB_INIT ( FAB = logfab,
: 1735      P 2448 2 |             FNS = .logfile [dsc$w_length],
: 1736      P 2449 2 |             FNA = .logfile [dsc$a_pointer],
: 1737      P 2450 2 |             FAC = PUT,
```

!Descriptor of log record

!Space for HELP\$LOG resultant filename
!Descriptor for result name
!NAM block for HELP\$LOG
!FAB for output to HELP\$LOG
!RAB for output to HELP\$LOG

!Descriptor for HELP\$LOG translation

! Initialize logical name

! Initialize descriptor for logical
! name translation

! Initialize name block

! Initialize fab


```
: 1738 P 2451 2 FOP = CIF,  
: 1739 P 2452 2 RAT = CR,  
: 1740 2453 2 NAM = lognam);  
: 1741 2454 2  
: 1742 P 2455 2 $RAB_INIT ( RAB = lograb, ! Initialize rab  
: 1743 P 2456 2 FAB = logfab,  
: 1744 2457 2 ROP = EOF);  
: 1745 2458 2  
: 1746 2459 2  
: 1747 2460 2 ! If HELP$LOG can be successfully translated,  
: 1748 2461 2 1. Create the file if it doesn't already exist.  
: 1749 2462 2 2. Connect to that file.  
: 1750 2463 2 3. Write the record to that file.  
: 1751 2464 2 4. Clean up afterwards.  
: 1752 2465 2  
: 1753 2466 2  
: 1754 2467 4 IF ((status = $TRNLOG (LOGNAM = logfile, RSLBUF = logfiletrn))  
: 1755 2468 3 AND (.status NEQ SS$_NOTRAN)) THEN IF (status = $CREATE (FAB = logfab))  
: 1756 2469 3 THEN BEGIN  
: 1757 2470 4 IF (status = $CONNECT (RAB = lograb))  
: 1758 2471 4 THEN BEGIN  
: 1759 2472 4 lograb [rab$_rsz] = .logdesc [dsc$_length];  
: 1760 2473 4 lograb [rab$_rbf] = .logdesc [dsc$_pointer];  
: 1761 2474 4 logrsdesc [dsc$_length] = .lognam [nam$_brsl];  
: 1762 2475 4 logrsdesc [dsc$_pointer] = .lognam [nam$_rsa];  
: 1763 2476 4 $PUT (RAB = lograb);  
: 1764 2477 5 IF NOT (status = $DISCONNECT (RAB = lograb))  
: 1765 2478 4 THEN SIGNAL ((shr$_closeout OR hlp$_facility OR sts$_k_warning),  
: 1766 2479 4 1, logrsdesc, .status, .lograb [rab$_stv]);  
: 1767 2480 3 END;  
: 1768 2481 4 IF NOT (status = $CLOSE (FAB = logfab))  
: 1769 2482 3 THEN SIGNAL ((shr$_closeout OR hlp$_facility OR sts$_k_warning),  
: 1770 2483 3 1, logrsdesc, .status, .lograb [rab$_stv]);  
: 1771 2484 2 END;  
: 1772 2485 2 RETURN true  
: 1773 2486 1 END;
```

!Of nohelp_log

```
.PSECT $OWNS,NOEXE,2  
08 000AB .BLKB 1  
08 000AC LOGSTRING:  
47 4F 4C 24 50 4C 45 48 000AD .BYTE 8  
          .ASCII \HELP$LOG\  
          .EXTRN SYSS$CREATE, SYSS$CONNECT  
          .EXTRN SYSS$PUT, SYSS$DISCONNECT  
          .EXTRN SYSS$CLOSE  
          .PSECT $CODE$,NOWRT,2  
007C 00000 NOHELP_LOG:  
56 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6 : 2395  
5E FDF4 CE 9E 00009 .MOVAB LIB$SIGNAL, R6 :  
08 AE 0000' CF 9B 0000E .MOVAB -524(SP), SP :  
          .MOVZBW LOGSTRING, LOGFILE : 2435
```


0060	8F	00	0C	AE	0000'	CF	9E	00014	MOVAB	LOGSTRING+1, LOGFILE+4	2436
			6E	FF	8F	9B	0001A	MOVZBW	#255, LOGFILETRN	2438	
			04	AE	FF00	CD	9E	0001E	MOVAB	LOGRESULT, LOGFILETRN+4	2439
			6E		00	2C	00024	MOVCS	#0, (SP), #0, #96, \$RMS_PTR	2445	
			00A4	CE	00A4	CE					
			6002	8F	80	0002E					
			00A6	CE	01	8E	00035	MOVW	#24578, \$RMS_PTR		
			00A8	CE	FF00	CD	9E	0003A	MNEGB	#1, \$RMS_PTR+2	
			00AE	CE		01	8E	00041	MOVAB	LOGRESULT, \$RMS_PTR+4	
0050	8F	00	00B0	CE	FF00	CD	9E	00046	MNEGB	#1, \$RMS_PTR+10	
			6E		00	2C	0004D	MOVAB	LOGRESULT, \$RMS_PTR+12		
			54	AE	54	AE	00054	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	2453	
			54	AE	5003	8F	80	00056	MOVW	#20483, \$RMS_PTR	
			58	AE	02000000	8F	D0	0005C	MOVL	#33554432, \$RMS_PTR+4	
			6A	AE		01	90	00064	MOVB	#1, \$RMS_PTR+22	
			72	AE	0202	8F	80	00068	MOVW	#514, \$RMS_PTR+30	
			7C	AE	00A4	CE	9E	0006E	MOVAB	LOGNAM, \$RMS_PTR+40	
			0080	CE	0C	AE	D0	00074	MOVL	LOGFILE+4, \$RMS_PTR+44	
0044	8F	00	0088	CE	08	AE	90	0007A	MOVB	LOGFILE, \$RMS_PTR+52	
			6E		00	2C	00080	MOVCS	#0, (SP), #0, #68, \$RMS_PTR	2457	
			10	AE	10	AE	00087				
			14	AE	4401	8F	80	00089	MOVW	#17409, \$RMS_PTR	
			4C	AE	0100	8F	3C	0008F	MOVZWL	#256, \$RMS_PTR+4	
					54	AE	9E	00095	MOVAB	LOGFAB, \$RMS_PTR+60	
					7E	7C	0009A	CLRQ	-(SP)	2467	
					7E	D4	0009C	CLRL	-(SP)		
					0C	AE	9F	0009E	PUSHAB	LOGFILETRN	
					7E	D4	000A1	CLRL	-(SP)		
					1C	AE	9F	000A3	PUSHAB	LOGFILE	
			00000000G	00	06	FB	000A6	CALLS	#6, SYS\$TRNLOG		
			52		50	D0	000AD	MOVL	R0, STATUS		
			09		52	E9	000B0	BLBC	STATUS, 1\$		
00000629	8F				52	D1	000B3	CMPL	STATUS, #1577	2468	
					03	12	000BA	BNEQ	2\$		
					008D	31	000BC	BRW	4\$		
					54	AE	9F	000BF	PUSHAB	LOGFAB	
			00000000G	00	01	FB	000C2	CALLS	#1, SYSS\$CREATE		
			52		50	D0	000C9	MOVL	R0, STATUS		
			7D		52	E9	000CC	BLBC	STATUS, 4\$		
					10	AE	9F	000CF	PUSHAB	LOGRAB	2470
			00000000G	00	01	FB	000D2	CALLS	#1, SYSS\$CONNECT		
			52		50	D0	000D9	MOVL	R0, STATUS		
			49		52	E9	000DC	BLBC	STATUS, 3\$		
			50		04	AC	D0	000DF	MOVL	LOGDESC, R0	2472
			32	AE	60	80	000E3	MOVW	(R0), LOGRAB+34		
			38	AE	04	A0	D0	000E7	MOVL	4(R0), LOGRAB+40	2473
			0104	CE	00A7	CE	9B	000EC	MOVZBW	LOGNAM+3, LOGRSDESC	2474
			FEFC	CD	00A8	CE	D0	000F3	MOVL	LOGNAM+4, LOGRSDESC+4	2475
					10	AE	9F	000FA	PUSHAB	LOGRAB	2476
00000000G	00				01	FB	000FD	CALLS	#1, SYSS\$PUT		
					10	AE	9F	00104	PUSHAB	LOGRAB	2477
00000000G	00				01	FB	00107	CALLS	#1, SYSS\$DISCONNECT		
			52		50	D0	0010E	MOVL	R0, STATUS		
			14		52	E8	00111	BLBS	STATUS, 3\$		
					1C	AE	DD	00114	PUSHL	LOGRAB+12	2479
					52	DD	00117	PUSHL	STATUS		
			FEF8	CD	9F	00119		PUSHAB	LOGRSDESC	2478	

LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine nohelp_log

F 1
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 60
(16)

		01	DD	0011D	PUSHL	#1	:
	00761058	8F	DD	0011F	PUSHL	#7737432	:
66		05	FB	00125	CALLS	#5, LIB\$SIGNAL	:
	54	AE	9F	00128	PUSHAB	LOGFAB	: 2481
00000000G	00	01	FB	0012B	CALLS	#1, SYSS\$CLOSE	:
	52	50	DO	00132	MOVL	R0, STATUS	:
	14	52	E8	00135	BLBS	STATUS, 4\$:
		1C	AE	DD	PUSHL	LOGRAB+12	: 2483
		52	DD	0013B	PUSHL	STATUS	:
	FEF8	CD	9F	0013D	PUSHAB	LOGRSDESC	: 2482
		01	DD	00141	PUSHL	#1	:
	00761058	8F	DD	00143	PUSHL	#7737432	:
66		05	FB	00149	CALLS	#5, LIB\$SIGNAL	:
	50	01	DO	0014C	MOVL	#1, R0	: 2485
		04	0014F	4\$:	RET		: 2486

; Routine Size: 336 bytes, Routine Base: \$CODE\$ + 0D5F


```
: 1775      2487 1 %SBTIL 'Routine remove_last_key';
: 1776      2488 1 ROUTINE remove_last_key (stringdescr, last_key_length) =
: 1777      2489 2 BEGIN
: 1778      2490 2
: 1779      2491 2 |+-
: 1780      2492 2 | FUNCTIONAL DESCRIPTION:
: 1781      2493 2 |
: 1782      2494 2 |     Remove the last keyword in the supplied string descriptor.
: 1783      2495 2 |
: 1784      2496 2 | INPUTS:
: 1785      2497 2 |
: 1786      2498 2 |     stringdescr = address of string descriptor for input text string
: 1787      2499 2 |
: 1788      2500 2 |     last_key_length = the length of the key that is to be removed
: 1789      2501 2 |
: 1790      2502 2 | OUTPUTS:
: 1791      2503 2 |
: 1792      2504 2 |     stringdescr = input descriptor with last key removed
: 1793      2505 2 |
: 1794      2506 2 | ROUTINE VALUE:
: 1795      2507 2 |
: 1796      2508 2 |     Always true.
: 1797      2509 2 |
: 1798      2510 2 | --
: 1799      2511 2 |
: 1800      2512 2 MAP
: 1801      2513 2 |     stringdescr : REF BBLOCK;
: 1802      2514 2 |
: 1803      2515 2 LOCAL
: 1804      2516 2 |     last_char;
: 1805      2517 2 |
: 1806      2518 2 | last_char = .stringdescr [dsc$w_length]
: 1807      2519 2 |     + .stringdescr [dsc$a_pointer] - 12
: 1808      2520 2 |     - .last_key_length;
: 1809      2521 2 |
: 1810      2522 2 | CH$MOVE (.subtopic [0], subtopic [1], .last_char + 1);
: 1811      2523 2 | stringdescr [dsc$w_length] = .last_char + 1
: 1812      2524 2 |     - .stringdescr [dsc$a_pointer];
: 1813      2525 2 |
: 1814      2526 2 RETURN true;
: 1815      2527 1 END;
```

! Of remove_last_key

00FC 0000 REMOVE_LAST KEY:

			57	04	AC	D0	00002	.WORD	Save R2,R3,R4,R5,R6,R7	: 2488
			56		67	3C	00006	MOVL	STRINGDESCR, R7	: 2518
			56	04	A7	C0	00009	MOVZWL	(R7), R6	: 2519
			56	08	AC	C2	0000D	ADDL2	4(R7), R6	
			56		0C	C2	00011	SUBL2	LAST_KEY_LENGTH, R6	: 2520
			50	0000'	CF	9A	00014	SUBL2	#12, LAST_CHAR	
			CF		50	28	00019	MOVZBL	SUBTOPIC, R0	: 2522
01	A6	0000'	56	04	A7	C2	00020	MOVC3	R0, SUBTOPIC+1, 1(LAST_CHAR)	
	67		56		0B	A1	00024	SUBL2	4(R7), R6	: 2524
								ADDW3	#11, R6, (R7)	:

LBR_OUTPUTHELP
V04=000

Prompting and library searching help function
Routine remove_last_key

H 1
16-Sep-1984 02:04:00
14-Sep-1984 12:37:45

VAX-11 Bliss-32 V4.0-742
[LBR.SRC]OUTPUTHLP.B32;1

Page 62
(17)

```
01  D0  00028
      04  0002B
```

```

MOVL    #1, R0
RET

```

: 2526
: 2527

; Routine Size: 44 bytes, Routine Base: \$CODE\$ + 0EAF


```
: 1817 2528 1 %SBTTL 'Routine remove_terminator';
: 1818 2529 1 ROUTINE remove_terminator (stringdescr) =
: 1819 2530 2 BEGIN
: 1820 2531 2
: 1821 2532 2 !++
: 1822 2533 2 FUNCTIONAL DESCRIPTION:
: 1823 2534 2
: 1824 2535 2 Remove the termination characters at the end of the string descriptor.
: 1825 2536 2
: 1826 2537 2 INPUTS:
: 1827 2538 2
: 1828 2539 2 stringdescr = address of string descriptor for input text string
: 1829 2540 2
: 1830 2541 2 OUTPUTS:
: 1831 2542 2
: 1832 2543 2 stringdescr = input descriptor with termination characters removed
: 1833 2544 2
: 1834 2545 2 ROUTINE VALUE:
: 1835 2546 2
: 1836 2547 2 Always true.
: 1837 2548 2
: 1838 2549 2 !--
: 1839 2550 2
: 1840 2551 2 MAP
: 1841 2552 2 stringdescr : REF BBLOCK;
: 1842 2553 2
: 1843 2554 3 WHILE (CH$RCHAR (.stringdescr [dsc$a_pointer]
: 1844 2555 3 + .stringdescr [dsc$w_length] - 1) LSS %X'20') ! While termination character present
: 1845 2556 3 AND (.stringdescr [dsc$w_length] GTR 0)
: 1846 2557 2 DO stringdescr [dsc$w_length] = .stringdescr [dsc$w_length] - 1; ! Remove terminator
: 1847 2558 2
: 1848 2559 2 RETURN true;
: 1849 2560 1 END; ! Of remove_terminator
```

0000 00000 REMOVE_TERMINATOR:

51	04	AC	D0	00002	.WORD	Save nothing	: 2529
50	04	BC	3C	00006 1\$:	MOVL	STRINGDESCR, R1	: 2554
50	04	A1	C0	0000A	MOVZWL	@STRINGDESCR, R0	: 2555
20	FF	A0	91	0000E	ADDL2	4(R1), R0	:
		0A	1E	00012	CMPB	-1(R0), #32	:
	04	BC	B5	00014	BGEQU	2\$:
		05	13	00017	TSTW	@STRINGDESCR	: 2556
	04	BC	B7	00019	BEQL	2\$:
		E8	11	0001C	DECW	@STRINGDESCR	: 2557
50		01	D0	0001E 2\$:	BRB	1\$:
		04	00021		MOVL	#1, R0	: 2559
					RET		: 2560

; Routine Size: 34 bytes, Routine Base: \$CODE\$ + 0EDB


```
: 1851 2561 1 %SBTTL 'Routine make_upper_case';
: 1852 2562 1 ROUTINE make_upper_case (idesc: REF BLOCK[, BYTE], oname: REF VECTOR[, BYTE]) =
: 1853 2563 2 BEGIN
: 1854 2564 2
: 1855 2565 2 ++
: 1856 2566 2 FUNCTIONAL DESCRIPTION:
: 1857 2567 2
: 1858 2568 2     Upper case the name described by string descriptor idesc and
: 1859 2569 2     put the name at location oname. (Also substitutes a space
: 1860 2570 2     character for a horizontal tab.)
: 1861 2571 2
: 1862 2572 2 INPUTS:
: 1863 2573 2
: 1864 2574 2     idesc =          address of string descriptor for input text string
: 1865 2575 2
: 1866 2576 2     oname =          address of buffer to contain uppercase output string
: 1867 2577 2
: 1868 2578 2 OUTPUTS:
: 1869 2579 2
: 1870 2580 2     oname : as described above
: 1871 2581 2
: 1872 2582 2 ROUTINE VALUE:
: 1873 2583 2
: 1874 2584 2     Always true.
: 1875 2585 2
: 1876 2586 2 --
: 1877 2587 2
: 1878 2588 2 LITERAL
: 1879 2589 2     fill= 0;
: 1880 2590 2
: 1881 2591 2 OWN
: 1882 2592 2     upcase_table: VECTOR [256, BYTE] INITIAL (BYTE (
: 1883 2593 2 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 240, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 254, 255)); ! HT -> space.
: 1884 2594 2 ! a-o -> A-O.
: 1885 2595 2 ! p-z -> P-Z.
: 1886 2596 2
: 1887 2597 2
: 1888 2598 2
: 1889 2599 2
: 1890 2600 2
: 1891 2601 2
: 1892 2602 2
: 1893 2603 2
: 1894 2604 2
: 1895 2605 2
: 1896 2606 2
: 1897 2607 2
: 1898 2608 2
: 1899 2609 2
: 1900 2610 2 CH$TRANSLATE (upcase_table,
: 1901 2611 2     .idesc[dsc$w_length], .idesc[dsc$a_pointer], fill,
: 1902 2612 2     .idesc[dsc$w_length], .oname);
: 1903 2613 2
: 1904 2614 2 RETURN true
: 1905 2615 1 END;                                !Of make_uppercase
```


														.PSECT \$OWNS,NOEXE,2	
														.BLKB 3	
														UPCASE_TABLE: .BYTE	
0E	0D	0C	0B	0A	20	08	07	06	05	04	03	02	01	00	000B5
1D	1C	1B	1A	19	18	17	16	15	14	13	12	11	10	0F	000B8
2C	2B	2A	29	28	27	26	25	24	23	22	21	20	1F	1E	000C7
3B	3A	39	38	37	36	35	34	33	32	31	30	2F	2E	2D	000D6
4A	49	48	47	46	45	44	43	42	41	40	3F	3E	3D	3C	000E5
59	58	57	56	55	54	53	52	51	50	4F	4E	4D	4C	4B	000F4
48	47	46	45	44	43	42	41	60	5F	5E	5D	5C	5B	5A	00103
57	56	55	54	53	52	51	50	4F	4E	4D	4C	4B	4A	49	00112
86	85	84	83	82	81	80	7F	7E	7D	7C	7B	7A	79	78	00121
95	94	93	92	91	90	8F	8E	8D	8C	8B	8A	89	88	87	00130
A4	A3	A2	A1	A0	9F	9E	9D	9C	9B	9A	99	98	97	96	0013F
B3	B2	B1	B0	AF	AE	AD	AC	AB	AA	A9	A8	A7	A6	A5	0014E
C2	C1	C0	BF	BE	BD	BC	BB	BA	B9	B8	B7	B6	B5	B4	0015D
D1	D0	CF	CE	CD	CC	CB	CA	C9	C8	C7	C6	C5	C4	C3	0016C
CO	DF	DE	DD	DC	DB	DA	D9	D8	D7	D6	D5	D4	D3	D2	0017B
CF	CE	CD	CC	CB	CA	C9	C8	C7	C6	C5	C4	C3	C2	C1	0018A
FE	DD	DC	DB	DA	D9	D8	D7	D6	D5	D4	D3	D2	D1	F0	00199
														FF	001A8
															001B7

.PSECT \$CODE\$,NOWRT,2

003C 00000 MAKE_UPPER_CASE:

.WORD Save R2,R3,R4,R5

MOVL IDESC, R0

MOVTC (R0), @4(R0), #0, UPCASE_TABLE, (R0), -

@ONAME

MOVL #1, R0

RET

; 2562

; 2611

; 2612

; 2614

; 2615

; Routine Size: 21 bytes, Routine Base: \$CODE\$ + 0EFD

; 1906 2616 0 END ELUDOM

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	440	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	3858	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	127	1	581	00:01.0

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD, INITIAL, OPTIMIZE)/LIS=LIS\$:OUTPUTHLP/OBJ=OBJ\$:OUTPUTHLP MSRC\$:OUTPUTHLP/UPDATE=(ENH\$:OUTPUTHLP)

: Size: 3858 code + 440 data bytes
: Run Time: 01:15.5
: Elapsed Time: 02:32.9
: Lines/CPU Min: 2079
: Lexemes/CPU-Min: 27990
: Memory Used: 351 pages
: Compilation Complete

0199 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

OLDLIB
LIS

OPENCLOSE
LIS

OUTPUTLP
LIS

LBRMSG
LIS

0200 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TRANSFER
LIS

DATABASE
LIS

PUTCACHE
LIS

LIBRAR

PREFIX
REQ

LIBRARIAN
MAP

CROSS
LIS

SUBS
LIS

PADLBR
LIS

COMPRESS
LIS

LIB
MDL

FILEIO
LIS

EXTRACT
LIS

DELETE
LIS